AD-A094 598 NAVAL POSTGRADUATE SCHOOL MONTEREY CA F/G 17/2

PARAMETRIC SIMULATION OF TACTICAL SINGLE CHANNEL FREQUENCY MODU--ETC(U

SEP 80 P A OLSON

UNCLASSIFIED NL

1 OF 2

AD A094598

AD A094598

NAVAL POSTGRADUATE SCHOOL

Monterey, California

LEVEL

DTIC
ELECTE
FEB 5 1981
S D
F

THESIS

Parametric Simulation
of
Tactical Single Channel
Frequency Modulated Communications

by

Philip A. Olson Jr.

September 1980

Thesis Advisor: A. L. Schoenstadt

Approved for public release; distribution unlimited.

DDC FILE COPY

81 2 04 017

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A094598	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Parametric Simulation of Tactical Single Channel Frequency Modulated Communications		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis (Sep 80)
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Philip Andrew Olson Junior		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		12. REPORT DATE September 1980
		13. NUMBER OF PAGES 148
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Naval Postgraduate School Monterey, California 93940		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

STAR
Tactical Communications
Communications Simulation
Movement Decision Logic Communications

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This thesis presents a stochastic simulation model of single channel FM communications that is designed to be used in conjunction with the Simulation of Tactical Alternative Responses (STAR) Combat Model. The communications modelled, the assumptions made, and the interface requirements necessary for inclusion of the Communications Model in the STAR Combat Model are explained in detail. The computer

code that is used to execute the Communications Model is included in the appendices. An overview is given of those portions of the basic STAR Model with which the Communications Model interfaces directly to provide the reader with sufficient background for the discussion of the Communications Model. The communications input requirements of the model, a definition of the purpose of each routine and event, and definitions of each global variable, set, and entity are provided so that this thesis can serve as a user's manual for the Communications Model.

Approved for Public Release: Distribution Unlimited

Parametric Simulation of Tactical Single Channel
Frequency Modulated Communications

by

Philip Andrew Olson, Junior
Major, United States Army
BS, Principia College, 1968

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL

September 1980

Author: Phil A Olson Jr

Approved by: Scott C Schmidt
Thesis Advisor

S H Parry
Second Reader

G. T. Howard, Acting
Chairman, Department of Operations Research

W M Woods
Dean of Information and Policy Sciences

ABSTRACT

This thesis presents a stochastic simulation model of single channel FM communications that is designed to be used in conjunction with the Simulation of Tactical Alternative Responses (STAR) Combat Model. The communications modelled, the assumptions made, and the interface requirements necessary for inclusion of the Communications Model in the STAR Combat Model are explained in detail. The computer code that is used to execute the Communications Model is included in the appendices. An overview is given of those portions of the basic STAR Model with which the Communications Model interfaces directly to provide the reader with sufficient background for the discussion of the Communications Model. The communications input requirements of the model, a definition of the purpose of each routine and event, and definitions of each global variable, set, and entity are provided so that this thesis can serve as a user's manual for the Communications Model.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENT

This implementation of communications for the movement decision logic in the STAR Combat Model utilizes a version of the communications model initially developed by Cpt. William Haislip and subsequently modified and revised by Associate Professor Arthur Schoenstadt.

The extensive computer resources required during development and testing of the model were ably provided by the staff of the W. R. Church Computer Center.

I also wish to express my heartfelt thanks to my wife, Linda, without whose dedicated support and nimble fingers this thesis probably would never have gotten through the typing phase.

I. INTRODUCTION

In order to accomplish its mission of winning the land battle, the Army must perform three basic functions: shooting, moving, and communicating. Therefore, it is reasonable to say that any combined arms combat model must portray these three functions with a commensurate degree of accuracy in order to realistically represent the combat process.

The question of precisely how to model communications between the elements, units, and individuals represented in a combined arms combat model is a critical one for the model designer. Communication must be an integral part of virtually every conceptualization of combat more complex than a one-on-one duel. As a result, communication is one of the most significant areas which the combat modeler must consider and with which he must deal effectively.

The assumptions made about the amount of communications allowed between elements, the time required for the communications to be completed, and the effect of communications (or lack thereof) frequently can be so far-reaching that they completely dominate many other effects which the modeler would like to study. They may also cause the model to be rejected as invalid. For this reason, the communications-related assumptions must be carefully thought out and implemented if the combat model is to be realistic and useful.

The original efforts in developing the STAR Combat Model were directed towards producing a high resolution Monte Carlo simulation of modern combined arms warfare. It was decided to base the terrain model on the work done by Needels for his thesis research in March 1976. He

demonstrated that bivariate normal distributions could be used to give a very good functional representation of continuous terrain. The STAR concept was to develop an event step simulation in which the impact point of each direct fire projectile was computed and every firepower activity of the represented elements was simulated insofar as possible. The movement of units, firing of weapons, and all of the related functions such as target acquisition, target selection, attrition, tactics and so on were modeled in considerable detail. Because the model was to focus on the evaluation of weapons and tactics, the original studies set aside the problem of communications for future study in order to get on with the development of the "shooting and movement" portions of the model. A description of the routines, events, and arrays of the STAR model which impact on the development of this thesis is provided in Appendix A.

The original work on the STAR model was done by Hagewood and Wallace in December 1978. At that time and stage of development the model portrayed a blue company under attack by a red battalion. Artillery and close air support/air defense modules were under development but were not included in the model at that time. A dynamic route selection module which considered enemy locations, terrain, and the tactical situation was being considered for inclusion in the model as was an improved module for target selection. In addition, fuel and ammunition resupply modules and a communications module were under development.

In March of 1977, four theses covering those areas mentioned above were written. The first, by Starner , developed a two-sided field artillery stochastic simulation which played both red and blue artillery at the battalion level. Provisions were made for upgrading the model to portray artillery at the brigade level when the brigade level was included in the STAR baseline.

The second, by Kramer, developed a deterministic simulation model for dynamic tactical route selection based upon the route selection technique used in the DYNTACS combat simulation model. Kramer's model was written in FORTRAN, and it was planned for eventual incorporation into the STAR model.

The third, by Broussard, developed a dynamic model for the tank commanders' target selection process based on eleven parameters fit by a regression model using responses from 128 tank gunners.

Finally, the fourth, by Haislip, developed a stand-alone communications model which will be discussed in a later paragraph.

In September of 1979, Caldwell and Meiers developed an air-to-ground and ground-to-air combined arms simulation designed for inclusion in the STAR model. At this time, STAR had been upgraded to include representation of a blue battalion under attack by a red regimental force; however, the production version which was used for several studies for TRADOC still did not include any of the modules such as the field artillery, dynamic route selection, dynamic target selection, or communications.

In his thesis work in March 1979, Haislip developed a model for single channel FM communications at the company and battalion level. Despite the fact that this code was written with the idea of incorporating it into STAR, there were several shortcomings which precluded its inclusion. First and most significant, the code was task specific so that in order to use it for the wide range of applications represented in the STAR model much of the logic and many of the variable and attribute names would have had to be altered. Second, the communications model only portrayed the communications for internal platoon fire coordination and field artillery fire direction. It was subsequently decided

that, for the near term, STAR would not explicitly model the communications below the company level for reasons which will be outlined in Chapter V. In spite of these shortcomings, the model written by Haislip did provide valuable insights into how to approach the problem of user-transparent communications modules for the higher echelons.

Since the purpose of this thesis was to design and implement a communications model for the movement decision logic within the framework of the existing STAR Combat Model, Haislip's work was used as the basis for the communications implementation with modifications and extensions in order to make the communications interface as robust as possible within the constraints imposed by the existing STAR organization. A description of the routines and events of the communications model is provided in Appendix B.

Because of the evolutionary nature of the STAR model there are always several "current" versions of the model in some stage of work. For the remainder of this thesis, any reference to STAR is a reference to the current production version of the model which does not include those modules under development. Also, references to the communications model refer to that module which was developed as a stand-alone model of the communications process using the decision logic flow from the production version of STAR. Hopefully, this communications module will be enhanced and included in one of the STAR baseline production models scheduled for completion later this year.

II. DESCRIPTION OF THE STAR MODEL

The STAR combined arms model is being developed at the Naval Postgraduate School to investigate the effectiveness of both tactical doctrine and materiel in a mid-intensity combined arms conflict. As previously noted, the model uses functionally generated continuous terrain and currently simulates a blue battalion under attack by an armor heavy red regimental force. The terrain currently in use simulates a 10 kilometer by 10 kilometer area near Fulda, Federal Republic of Germany; however, other terrain boxes have been developed including one 40 kilometers by 80 kilometers which is planned to be used with simulation of a blue brigade-level force under attack by a red division-level force.

The current version of STAR simulates individual weapons systems such as XM1 tanks, TOW's, BMP's, T72 tanks, and dragons. These entities detect enemy systems, select and acquire targets, load and fire their weapons, and move about the simulated battlefield based upon input parameters supplied by the model user. The user defines the types of weapon systems, the ammunition available, the tactics to be used for selecting and engaging targets, and the force ratio and range breakpoints used to determine if a unit wants to withdraw from its current position. A line-of-sight module is used to determine whether or not one system can see and, therefore, engage an enemy system based upon their respective locations on the continuous terrain.

Air defense, dismounted infantry, and artillery modules are under development for inclusion in the model. Also, preliminary work has been done on a vehicle evacuation and repair module to simulate one of the major logistics

functions.

During the initialization process for the STAR model, each company is assigned a weight based upon the users' view of the importance of the particular company in the overall defensive posture of the battalion. This value, COWT, is an attribute of the company for a particular phase line and is used by the movement decision logic. A company can and frequently does have different values of COWT for different phase lines. Also during the initialization, the user assigns a value to column 4 of the array TABLE for each company and weapon system on every phase line. This value represents the restriction status of the unit and must be a one or a zero. If the value is a one, the company or weapon system is restricted from moving from that phase line position unless given permission by its superior commander. If the value is zero, the company or weapon system can move at will. TABLE also stores codes for all of the tactical actions allowed for each company and weapon system on each phase line. Associated with each of these action codes, the TABLE array contains a force ratio and an attrition level which trigger the action to be taken. (Technical Report NPS55-79-023 by Parry and Kelleher contains a complete description of the parameters stored in array TABLE.)

The movement logic in STAR is handled by Routine DECISION which uses the unit number of the unit requesting or being ordered to move, the row number in the TABLE array of the weapon system to be moved by the unit, and the appropriate column number in the TABLE array. This information is combined with whether or not the weapon system is restricted from moving from its present phase line position, which is in column 4 of the TABLE array. A flow diagram of the logic in Routine DECISION is given in Figure 1. The arrays used by the movement decision logic are described in Appendix A.

STAR DECISION LOGIC

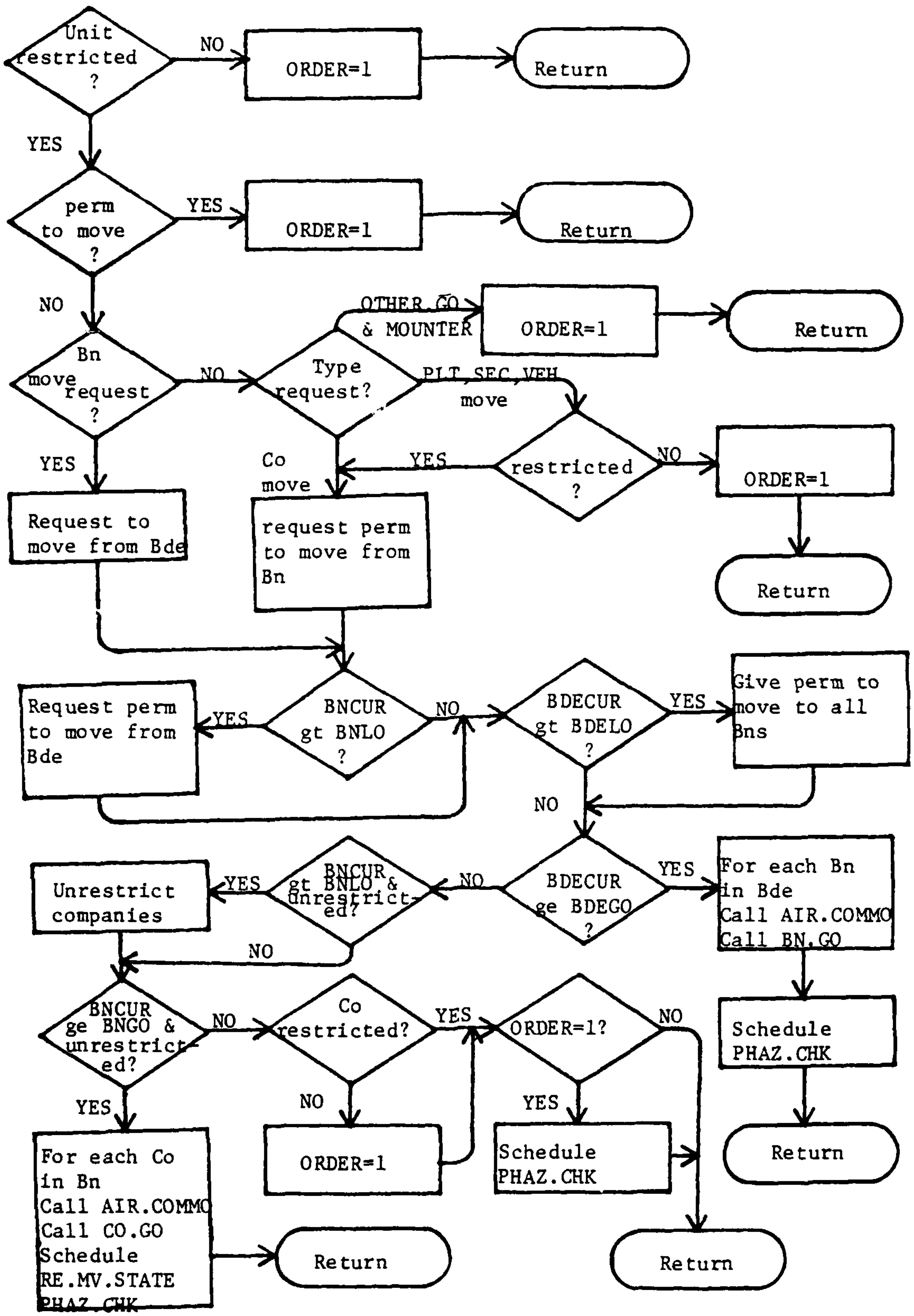


FIGURE 1

During simulation of a battle, blue forces may be allowed by the model users' choice of input parameters to move out of their current position as a result of heavy attrition by enemy forces or because of the range to the nearest enemy element. Each shot at a blue element causes routine BUG.CHK to check the range to the enemy. If it is less than the value supplied by the user in column three of the array TABLE for that phase line and that weapon system or company, then routine DECISION is called to determine if the weapon system or company is restricted or is allowed to move from its current position. Similarly, routine ACTION calls routine DECISION as blue systems are killed by enemy fire and force ratio decision points are reached.

In order to determine when a battalion will give its companies permission to move and when it will order its companies to move, each battalion has three attributes. Two of these, BNLO and BNGO, are assigned values by the user. The third, BNCUR, is set to zero when a battalion first occupies a phase line. Whenever a company requests permission to move, its weight is added to the BNCUR value. If this value exceeds the user defined BNLO value, the battalion gives all of its companies permission to move at will. If the value of BNCUR exceeds the user input value of BNGO, the battalion orders all of its companies to move. As long as BNCUR is less than BNLO, restricted companies and systems of that battalion stay restricted and are not given permission to move.

When routine DECISION is called, the restriction status of the company or weapon system in array TABLE is checked, and the BNCUR value is updated if required. Then, if the company or system is allowed to move, the appropriate movement routine is called to handle the movement process.

Since in the current version of STAR all of the checking of the company status, updating of the BNCUR value, and

calling of the movement routines is done in the same routine, an assumption is made of perfect and instantaneous communications from the company to the battalion and back to the company. This and the other assumptions of STAR dealing with the communications and decision processes are discussed in the next chapter.

III. COMMUNICATIONS ASSUMPTIONS OF STAR

The current STAR Combat Model makes several far-reaching assumptions concerning communications during simulated battles. These assumptions are driven by the fact that neither communications nor electronic warfare is modeled explicitly in the current production version of STAR. As noted in Chapter I, the modeling of these phenomena were left for later work by the original studies done on STAR. In order of decreasing importance the implicit assumptions made in the STAR model concerning communications and electronic warfare are listed and explained below.

1) Communications of information from one element to another takes no time. This assumption means that a primary effect of communications on the battle process, the time it takes to reach decisions and affect the actions or posture of remote units, has been ignored.
2) Communication between units is independent of the locations of the units and of the intervening terrain. In practice, the intervening terrain often precludes any communication between units.
3) All designated recipients of a communication receive exactly the same information at exactly the same time. This means that messages are never lost and all are handled with perfect efficiency.
4) The probability of successful communications between two elements is not affected by the combat situation nor by the kill condition of either element. This means that communications gear is never damaged during the battle to such an extent that it cannot be repaired by the operator, or replaced.

5) All methods of communications are equivalent since they all take no time and they all have the same probability of success.

6) All commanders are always instantaneously aware of a change in the status of any of their units.

Of these six assumptions, the one of paramount importance is the first because of its great effect on the outcome of the battle. This thesis focuses on implementing a communications model for the movement decision logic of the STAR model and on providing a method of modeling messages so that this first assumption can be eliminated in favor of one which deals only with the distribution of the time delays caused by transmission of messages from one entity to another.

A description of the problems caused by these assumptions is presented in the next chapter.

IV. DESCRIPTION OF THE PROBLEM

Because of the intense effort to implement a production version of STAR as an effective combined arms combat model, the development of the communications aspects of the model has not been the primary concern of the model developers. The predominant view concerning communications between units has been that the oversimplified portrayal would be adequate until such time as the model was sufficiently sophisticated to allow the communications to play a significant role in the exercise of the model.

The lack of an adequate communications model as outlined above together with the assumptions of the STAR model regarding communications which are stated in Chapter III have a great effect on the results of the model in certain critical situations. For example, since the loss of time in transmitting information is ignored, the battalion commander is instantly aware when one of his companies is being heavily attrited by the enemy and wants to withdraw from a particular position, and no time is taken up by the decision process which generates the order or the communication process which transmits the movement authorization to the company. This, in turn, means that this particular blue company is, on the average, attrited less than it would be if the company had to wait some length of time to get permission to move from its current position. Thus blue casualties are lower than they would be for a more realistic simulation of the decision and communications process.

This thesis work was begun at a time when it was decided that a more detailed treatment of communications was necessary. It was determined that this was an area of the

model where significant improvements in model realism could be made. At the same time, it was apparent early in the development of the communications model that major portions of the STAR code would have to be completely rewritten to implement a realistic communications simulation model. Many of the decisions which are based upon communicated information are "hard-wired" into the STAR code which means that in order to change these decisions, the code itself had to be radically altered to provide for more flexibility. As noted previously, this thesis work was begun with the goal of implementing a communications model within the STAR model which would handle the message flow for the movement decision logic. This model was to be sufficiently flexible and expandable to handle the communications requirements for other functions as they are converted to the explicit communications representation. A description of the assumptions of the communications model is provided in the next chapter.

V. ASSUMPTIONS OF THE COMMUNICATIONS MODULE

The communications model developed in this thesis makes the following assumptions.

1) A message on the communications net will not be interrupted by another message trying to get on the net. This means that there is no scheme for handling different message precedence levels.
2) Human error in interpretation of the messages is negligible.
3) Signal strength during transmission of a specific message does not vary significantly.
4) The probability of jamming a given message is independent of whether or not a previous message on the same net was jammed.
5) None of the unit commanders are killed during the course of the battle. This is accomplished by setting all the commanders ammo counts to zero and by setting each commander's ACVT.TYPE attribute equal to one.
6) Below the company level, communications by messenger takes approximately the same time as electrical communications and therefore communications at this level is assumed and is not modeled.

Some brief comments on these assumptions are in order and they are provided below.

Until such time as the communications model is sufficiently sophisticated to allow for modeling of traffic priority, the first assumption provides an adequate model for a process where all traffic is of the same relative importance.

The second assumption covers a multitude of faults which

exist in the real world communication process. However, just as it is not feasible to try to model the differences in viewpoint between the red and blue commanders except in terms of doctrine, it is not feasible to try to model the possible ways a message can be misinterpreted since this is a function of so many disparate factors.

The third assumption provides a good approximation to reality except during bad weather when atmospheric conditions change with great rapidity, and when jamming is encountered. This assumption has an explicit proviso, that is that the message lengths must be short relative to the time scale for atmospheric or equipment changes.

The fourth assumption is applicable to this version of the communications model but will undoubtedly be superceeded by later revisions which are anticipated to model the phenominon of jamming in great detail. In the current version the probability of jamming is set by the user by his choice of the values for the global variables which are used by the SIMSCRIPT random number generator. A description of the global variables used in the communications model is provided in Appendix C.

The fifth assumption is made because this entire area of battlefield synergism is not being played in the model. In real combat, commanders are killed and units lose or gain effectiveness based upon the capabilities of the subordinates who assume command. There is no data to support any sort of model of this gain/loss of effectiveness, and it is not anticipated that any attempt will be made to incorporate this sort of detail in the model since it turns primarily on capabilities of individuals such as leadership, judgment, and decisiveness which are essentially not quantifiable.

The rationale for the sixth assumption and for not modeling the communications below the company level is that

these emitters are generally so close together that it is almost always possible to pass traffic between them, and jamming them becomes prohibitively expensive in terms of level of effort and expenditure of resources. Also, because they are relatively close together, critical information can be passed by messenger in a time that is roughly equivalent to the time required to pass the information by electronic methods. This is not true for messages between the higher level commanders where the time for a messanger to deliver the information would be orders of magnitude larger than the time to deliver the information by electronic means. In addition, modeling all these details would consume valuable computer resources which can be more profitably used to represent other aspects of the combined arms battle.

Unlike the communications model currently employed by the STAR model which assumes no loss of time in transmitting information from one unit to another during the course of the battle, the one developed in this thesis provides the user of the model with almost complete freedom in his choice of how to model the distributions of the time delays involved in the communications and decision processes. The guidelines for model parameterization are covered in detail in Chapter VII. A complete description of the communications model is provided in the next chapter.

VI. DESCRIPTION OF THE COMMUNICATIONS MODULE

The communications model developed in this thesis is an amalgum of ideas and concepts from many sources. The logic of the STAR model necessarily makes many decisions which in real life are based upon information communicated to the commanders by sources outside their immediate area. Most of this information is provided to the commander in the form of a written or verbal message. While these messages do not conform to any set format, there are certain minimum essential elements of information contained in them which are determined by the type or intent of the message. For example, as a minimum, a request by a company for permission to move from a specified position must include a unit designator and some type of code to identify this message as a request to move. Since different types of messages will necessarily have different minimum information contents, it was decided that the messages and their text portions should be modeled separately. In order to help the reader with understanding the logic flow which will be discussed in this chapter, the entities and their pointers which are used by the communications logic are diagrammed in Figure 2. Also, the arrays and sets used by the communications logic are described in Appendices F and G respectively.

To represent the flow of information within the model, messages are generated as temporary entities whenever a decision point is reached where information must be passed between two subscribers which are spatially separated or are not from the same command level. There is one exception to this policy: the internal company communications. Communications from individuals to their platoon leaders and from the platoon leaders to the company commanders are

COMMUNICATIONS ENTITY AND POINTER DIAGRAM

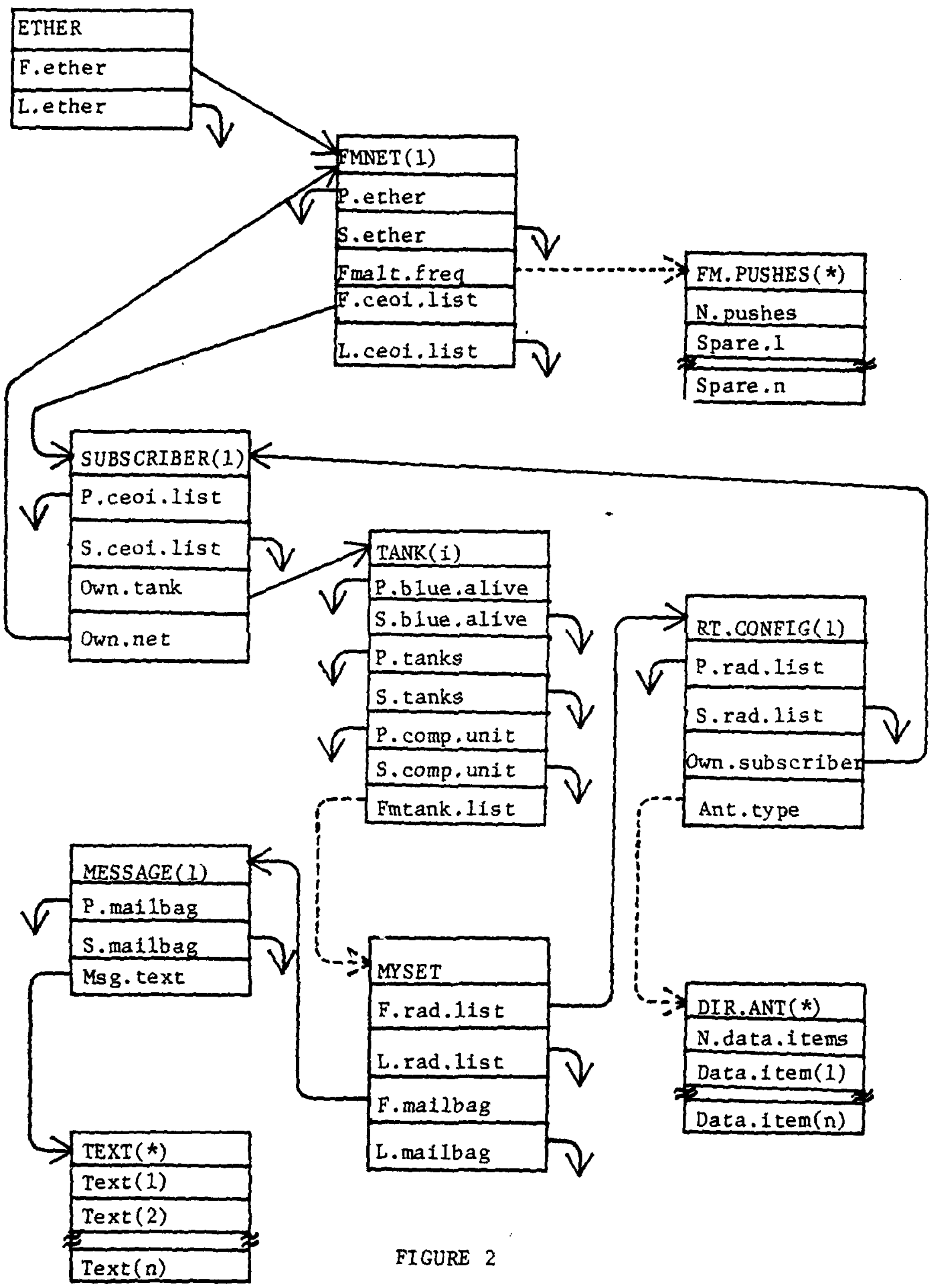


FIGURE 2

assumed to be 100% reliable and take no time. This is not the case for communications between the company commanders or for communications between any of the higher level commanders.

The communications module being implemented uses the same basic decision logic as that diagrammed in Figure 1. However, since the primary emphasis of this thesis was to model the communnications delays and the effects they have on the eventual outcome of the simulation, the DECISION routine had to be completely rewritten to allow the representation of these delays. Routine DECISION was actually taken apart and broken down into seven mutually disjoint new routines. These seven routines utilize the same arrays and sets as the original DECISION routine to model the decision process. Four of these routines, DEC.CO, DEC.BN, DEC.BDE, and DEC.DIV, handle the movement decision logic at the various command levels. The fifth routine, DECISION, acts as the driver which initiates the movement decision process when called by either BUG.CHK or ACTION from the STAR logic. The sixth routine, EXIT, provides an exit from the movement decision logic under specified conditions. The seventh routine, GEN.MOVE.DECISION.MSG, produces the appropriate message which is transferred between these routines carrying the information required for the decision to be made. These routines are discussed in more detail later in this chapter along with all the communications routines and events necessary for the transfer of messages between them. The SIMSCRIPT code for these modules is provided in Appendices H through O respectively. Instead of changing the information known by all the commanders at all levels when a company commander decides to request permission to move, the communications model stochastically generates and sends a MESSAGE from the company commander to the battalion commander containing the

request. When this MESSAGE transmission ends, the model provides the STAR decision logic with the information necessary for a decision by the battalion commander and then stochastically models the transmission of the decision by generating and sending a MESSAGE from the battalion commander to the company commander. If the users choice of inputs does not allow the battalion commander to give permission to move to the company commander, another MESSAGE is generated and sent. This new MESSAGE goes to the brigade level and when the transmission ends, the decision logic at the brigade level either generates the decision and sends a MESSAGE back to the battalion or passes a further request for permission to move on to the division level. As the model stands, the division level cannot send a message higher and thus is the ultimate decision authority. The user should be aware that his choices of break point bounds and restriction status for the simulated units are critical for a realistic simulation. Flow diagrams of the modified decision logic used in the communications model are given in Figures 3, 4, 5, 6, and 7. In addition, a partial listing of the output of the model is provided as Appendix II. This listing has been greatly shortened, but serves to illustrate the type of output which can be expected from the model during execution. A description of the flow of the communications model routines and events will be given next to familiarize the reader with the details of the model.

A. COM.MAIN

The SIMSCRIPT code for routine COM.MAIN is provided in Appendix D. It is the first of the communications model routines to be executed. It is called from BL.CREATE and it in turn calls all the initialization routines for the

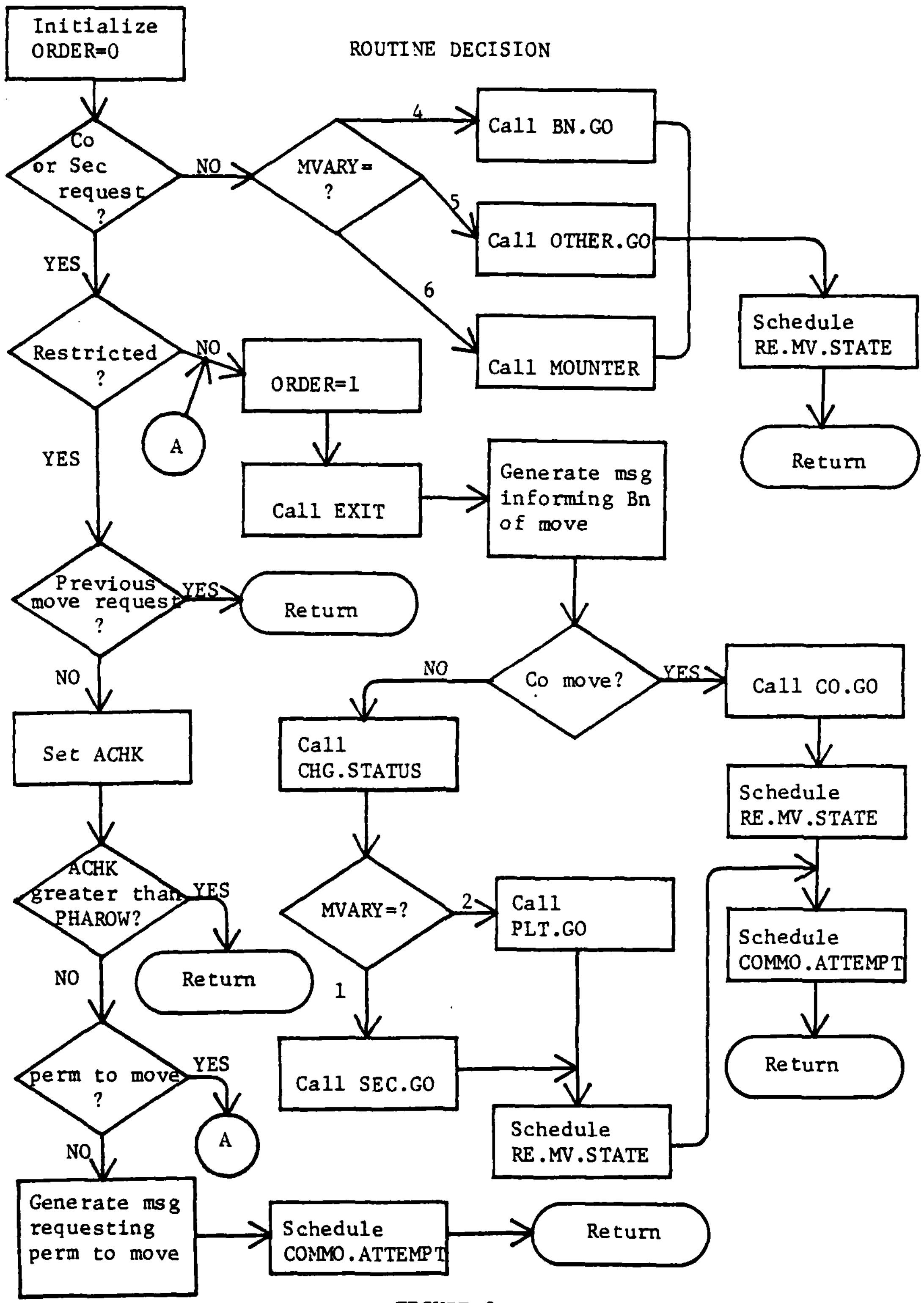

ROUTINE DECISION
Initialize ORDER=0
Co or Sec request ?
NO
MVARY= ?
4
Call BN.GO
5
Call OTHER.GO
6
Call MOUNTER
Schedule RE.MV.STATE
Return
YES
Restricted ?
NO
ORDER=1
A
Call EXIT
Generate msg informing Bn of move
YES
Previous move request ?
YES
Return
Co move?
NO
YES
Call CO.GO
NO
Set ACHK
Call CHG.STATUS
Schedule RE.MV.STATE
ACHK greater than PHAROW?
YES
Return
MVARY=?
2
Call PLT.GO
Schedule COMMO.ATTEMPT
NO
1
Return
perm to move ?
YES
A
Call SEC.GO
Schedule RE.MV.STATE
NO
Generate msg requesting perm to move
Schedule COMMO.ATTEMPT
Return

FIGURE 3

ROUTINE DEC.CO

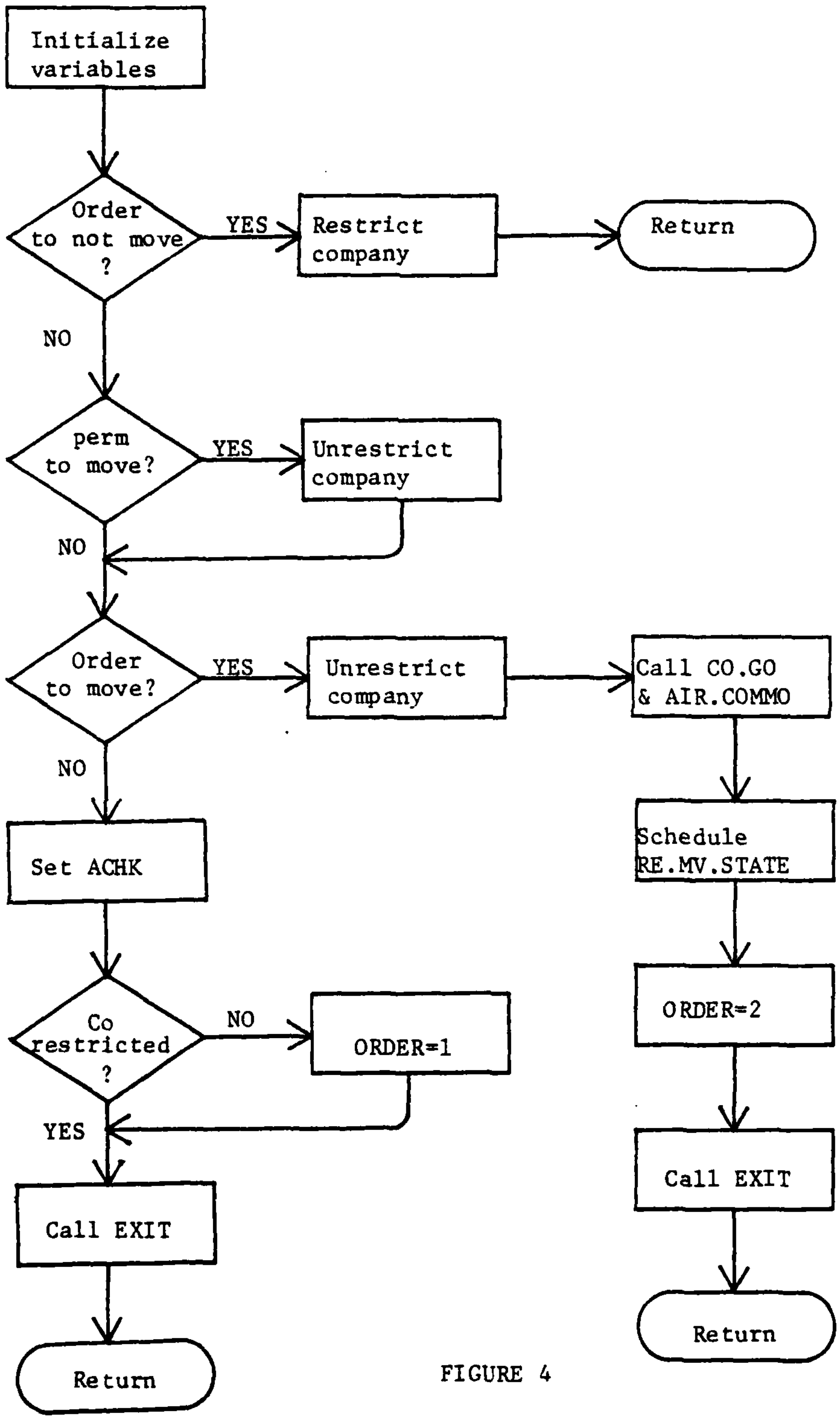


FIGURE 4

ROUTINE DEC.BN

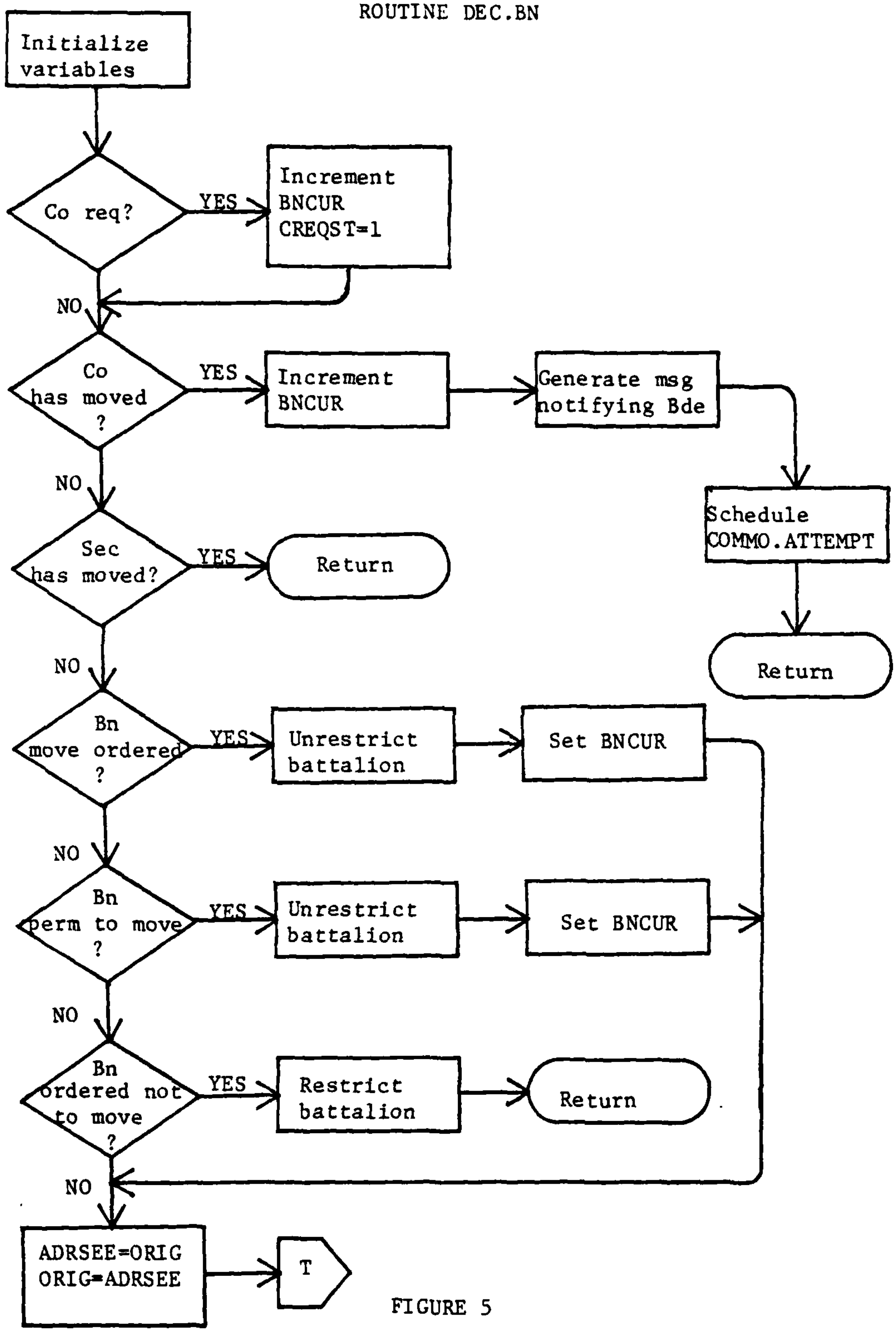


FIGURE 5

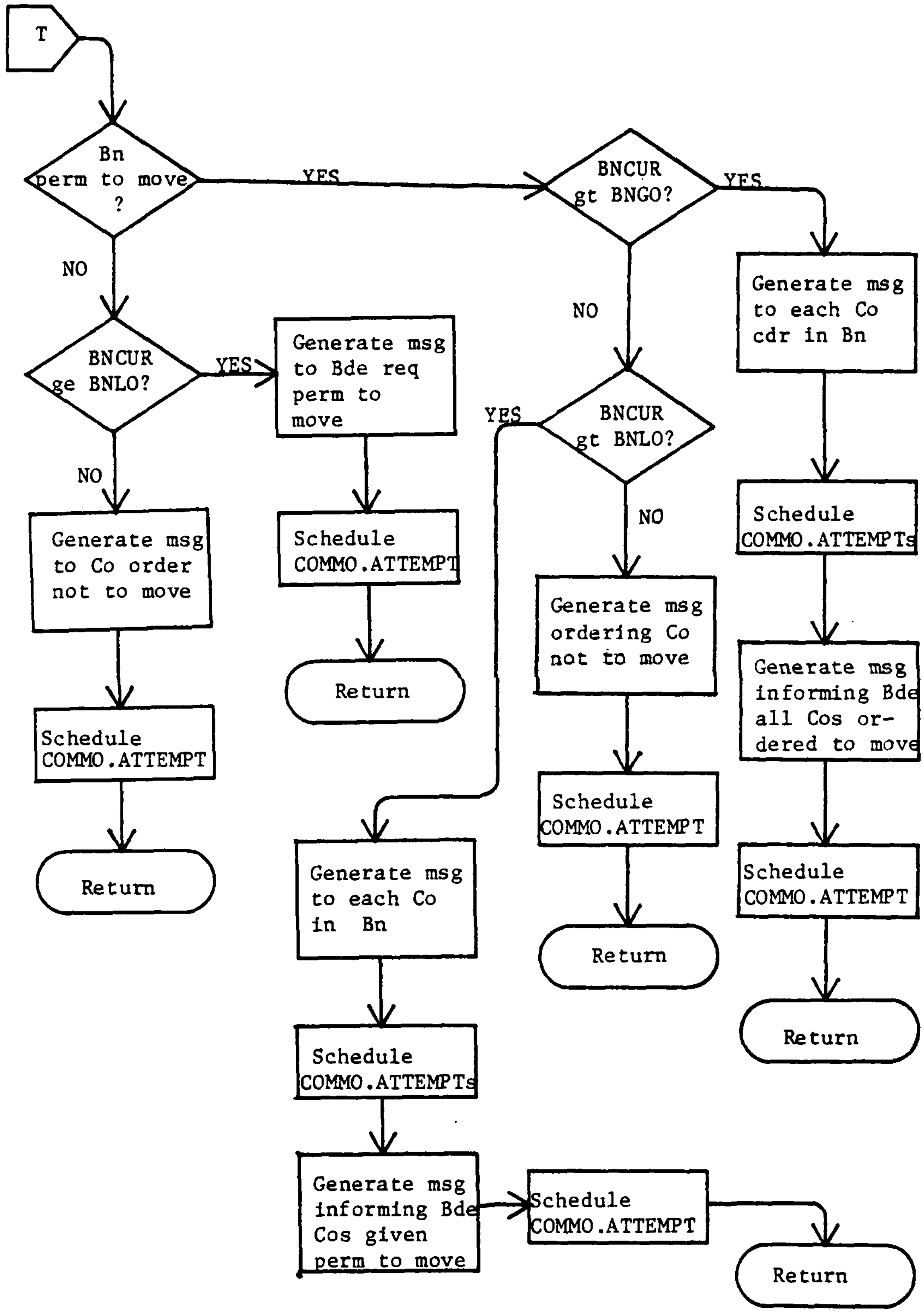


FIGURE 5 (cont.)

ROUTINE DEC.BDE

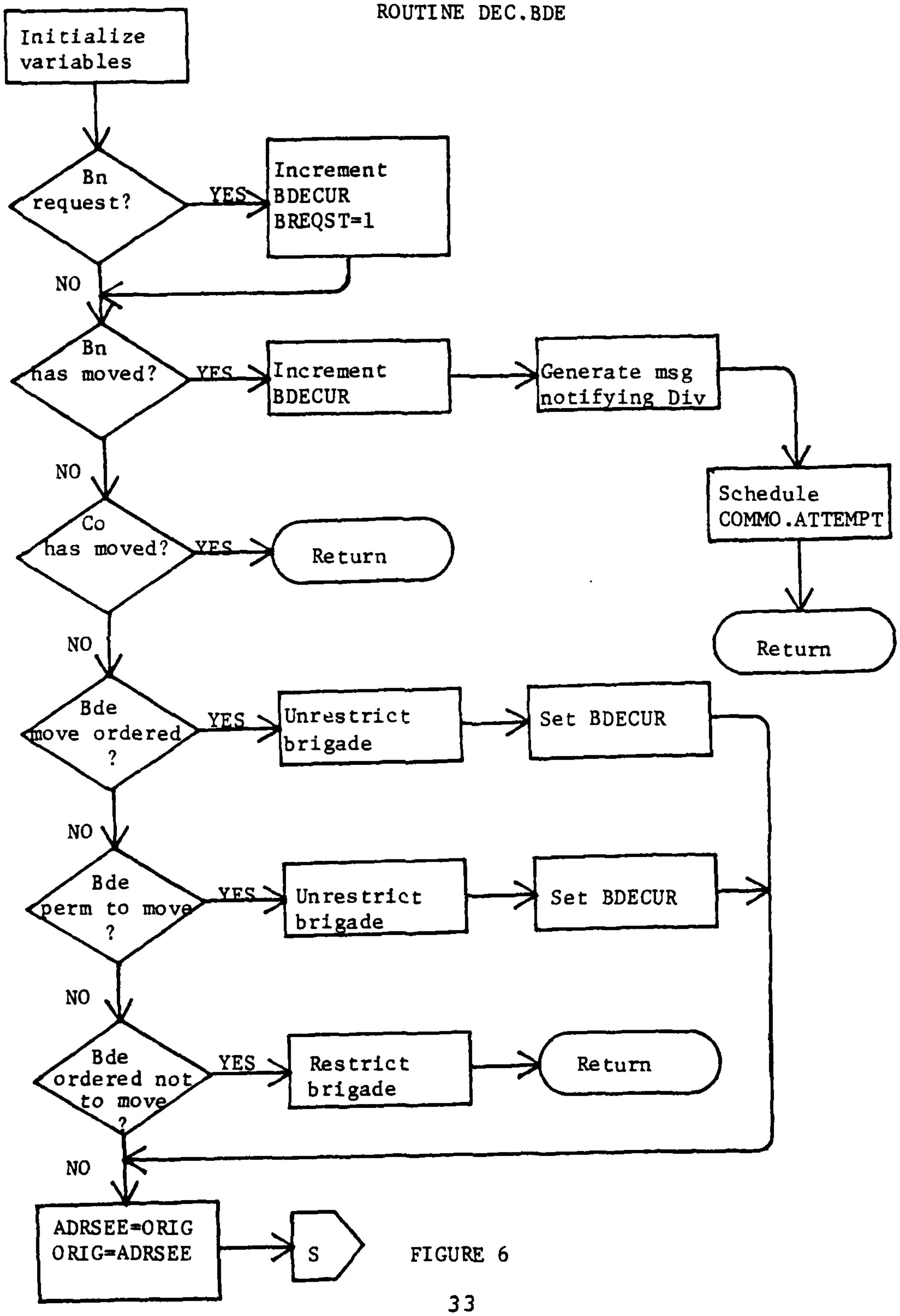


FIGURE 6

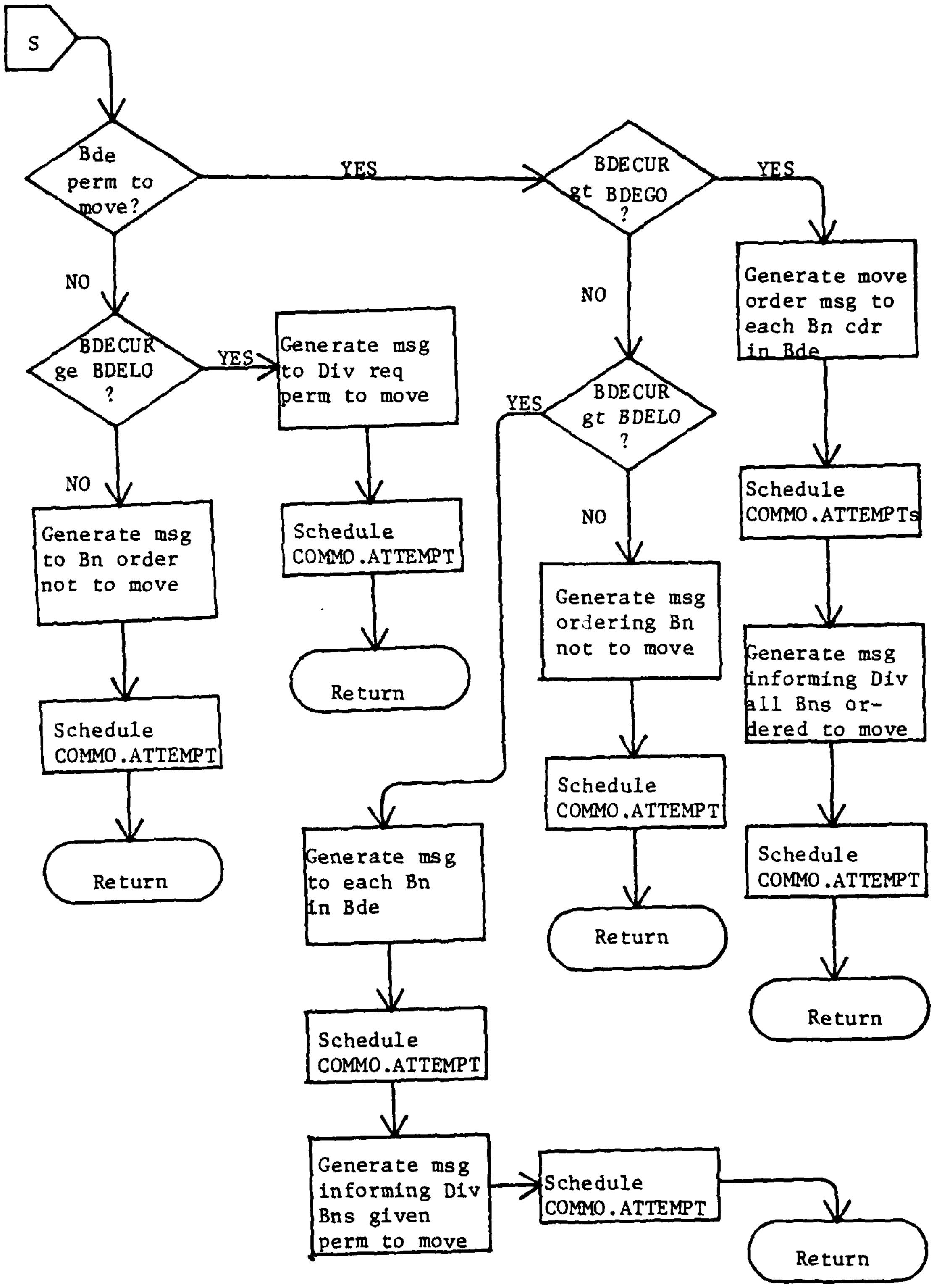


FIGURE 6 (cont.)

ROUTINE DEC.DIV

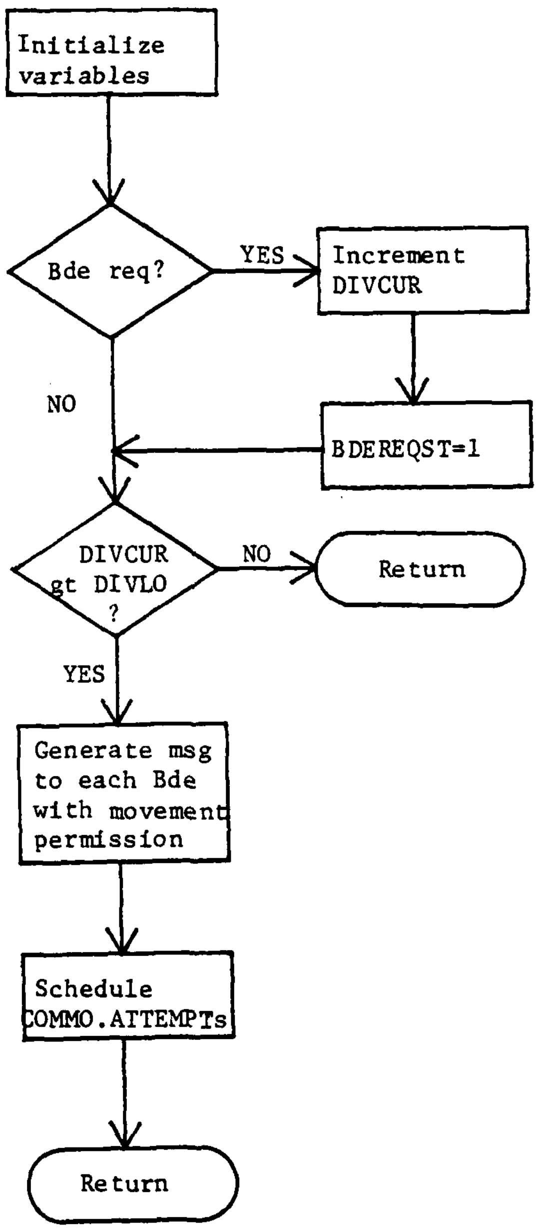


FIGURE 7

communications module before listing the attributes of each B.MVR.CDR, COMPANY.COMMANDER, BN.COMMANDER, BDE.COMMANDER, DIV.COMMANDER, and UNIT. COM.MAIN is the last executable routine called by BL.CREATE, and when it terminates, the SIMSCRIPT timing routine takes control since the simulation has already been started.

B. BUILDNETS

The SIMSCRIPT code for routine BUILDNETS is provided in Appendix P. It reads in the data on the FMNETs which are to be simulated and on which UNITs belong to each FMNET. It then creates the nets and the SUBSCRIBERs and places the SUBSCRIBERs in the CEOI.LIST for their net. It also reads any alternate frequencies which are assigned to the FMNET and stores them in the array FM.PUSHES. It then returns control to COM.MAIN.

C. CHECKNETS

The SIMSCRIPT code for routine CHECKNETS is provided in Appendix Q. It goes through each FMNET that has been created and prints out the information on them. It prints the net number, the net frequency, the number and frequency of any alternate frequencies, and the number of SUBSCRIBERs assigned to the net before returning control to COM.MAIN.

D. BNETTBL

The SIMSCRIPT code for routine BNETTBL is provided in Appendix R. It cycles through all the FMNETs and creates and

prints out two tables. The first, NETTBL, is used during execution of the communications routine. It contains the identity numbers of the SUBSCRIBERs belonging to the ith FMNET in the ith row. The second table has the address pointers to the SUBSCRIBERs in the ith FMNET in its ith row. After printing both tables, BNETTBL returns control to COM.MAIN.

E. ROUTINE COMMGEAR

The SIMSCRIPT code for routine COMMGEAR is provided in Appendix S. It reads in the data on how many UNITS/TANKS have communications equipment and creates a MYSET for each of them. It then creates the RT.CONFIGs for each TANK and reads in the characteristic data for each RT.CONFIG assigned to each TANK and files the information as attributes of the RT.CONFIG. If the data indicates that the antenna is directional or remoted, a message is printed and the job terminates since there are no provisions in the current model for handling directional or remoted antennas. Eventually, these are planned for inclusion, however. The routine then cycles through the FMNETs and sets the first message number for each SUBSCRIBER in each net equal to n001, where n is the identity number of the SUBSCRIBER. Finally, this routine places the SUBSCRIBERs in the nets and returns control to COM.MAIN.

F. ROUTINE COMMCHECK

The SIMSCRIPT code for routine COMMCHECK is provided in Appendix T. It cycles through each TANK in BLUE.ALIVE and prints out the communications gear assigned to each TANK.

For each RT.CONFIG it prints out the x,y, and z coordinates, the type of radio, the radio mode of operation, the antenna type, and the location of the antenna which currently must be the same as the location of the TANK. If either a directional or remoted antenna is detected, a message is printed and the program halts. This is done because the routines which are planned to handle this have not yet been included in the model. Finally, it prints the number of the net to which each RT.CONFIG is tuned before returning control to COM.MAIN.

G. ROUTINE INITIALIZE

The SIMSCRIPT code for routine INITIALIZE is provided in Appendix U. It reads in the values for the parameters used by the communications routines for generating random numbers and for controling the print out of data. It also reads in the lower and upper bounds for the destination codes of messages being sent to the company, battalion, brigade, and division levels. It also calls routines STAT.DUMP and SSTAT.DUMP before it returns control to COM.MAIN.

H. ROUTINE STAT.DUMP

The SIMSCRIPT code for routine STAT.DUMP is provided in Appendix V. It lists the attributes of each COMPANY.COMMANDER, BN.COMMANDER, BDE.COMMANDER, and DIV.COMMANDER and is used primarily as an aid to debugging. In addition to the initial call from routine INITIALIZE, it is called from routine END.XSMN periodically based upon the value supplied by the user for global variable COM.DATA.DUMP. Complete descriptions of the global variables

used in the communications model are provided in Appendix C while the attributes of the permanent and temporary entities are described in Appendices D and E respectively.

I. ROUTINE SSTAT.DUMP

The SIMSCRIPT code for routine SSTAT.DUMP is provided in Appendix W. It prints out the contents of the BDECORD, BNCORD, and COCORD arrays. This is another of the print routines designed to aid in debugging the code if it should fail. The arrays used in the communications model are listed and explained in Appendix F.

Once initialization of the modules has been accomplished and the SIMSCRIPT II.5 timing routine has taken control of the system, the event step sequence determines the order in which the routines and events will take place. The following description of the routines and events is done as if tracing the flow of a specific decision through the system. Whenever in the following narrative the phrase "a message is sent" is encountered, it should be taken to mean "a COMMO.ATTEMPT is scheduled."

J. ROUTINE DECISION

The SIMSCRIPT code for routine DECISION is provided in Appendix H. The flow diagram for the logic of routine DECISION is given in Figure 3. DECISION is called by either BUG.CHK or ACTION from the STAR logic. Both of these routines were significantly altered to allow them to interface with the new decision logic routines. Routine DECISION checks to see if this call is a company or section requesting permission to move by testing the value of the

element of array MVARY referred to by the row and column of the call. If it is not a company or section, DECISION calls the appropriate move routine and schedules a RE.MV.STATE for the calling unit in 60 seconds and then returns from routine DECISION. If it is a company or section request, the logic goes on to test the restriction status of the company or section which is located in the TABLE array. If the company or section is not restricted, the appropriate movement routine is called and a message is created and sent informing the battalion of the unit move. If the unit is restricted from moving, the value of CREQST is checked to see if the unit has previously requested permission to move. If a previous request has occurred, a message is printed noting that fact and then the program returns from routine DECISION. If there has been no previous request for permission to move, the logic checks to see if the unit is on the same phase line as the rest of the units of the battalion. If the unit is farther to the rear than the battalion, a message is printed, and the program returns from routine DECISION. If the unit is on the battalion phase line the program checks to see if the unit has been given permission to move. If the unit has been given permission to move, the appropriate movement routine is called and a message is generated and sent informing the battalion of the unit move. If the unit does not have permission to move, a message is generated and sent to the battalion requesting permission to move. After this, the program exits from routine DECISION.

K. ROUTINE GEN.MOVE.DECISION.MSG

The SIMSCRIPT code for routine GEN.MOVE.DECISION.MSG is provided in Appendix N. It is called by all the movement

decision logic routines and it creates the message which is sent from one level to another by the flow of the communications logic. This routine first creates a message and then sets up an array called TEXT. A pointer to the TEXT array where the message's content is stored is saved as an attribute of the MESSAGE. After storing all the relevant information either as attributes of the MESSAGE or in the TEXT array, the program returns from routine GEN.MOVE.DECISION.MSG.

L. EVENT COMMO.ATTEMPT

The SIMSCRIPT code for event COMMO.ATTEMPT is provided in Appendix X. The flow diagram for the logic of event COMMO.ATTEMPT is provided in Figure 8. Event COMMO.ATTEMPT takes the messages which are created by the decision process and stochastically models their transmission. This event first checks both the sending and receiving UNIT/TANK to insure that both have radios. If either one does not have a radio, it prints an error message and stops the simulation. It then checks to be sure that both UNITs are on the same FMNET. If they are not, it prints an error message and stops the simulation. If they are, and the net is not busy, it sets the net to busy and calls routine TECH.COMMO which checks to see if communication between the two radios is technically possible. If the net is busy, it files the MESSAGE in the MAILBAG for that SUBSCRIBER and returns from event COMMO.ATTEMPT.

M. ROUTINE TECH.COMMO

The SIMSCRIPT code for routine TECH.COMMO is provided in

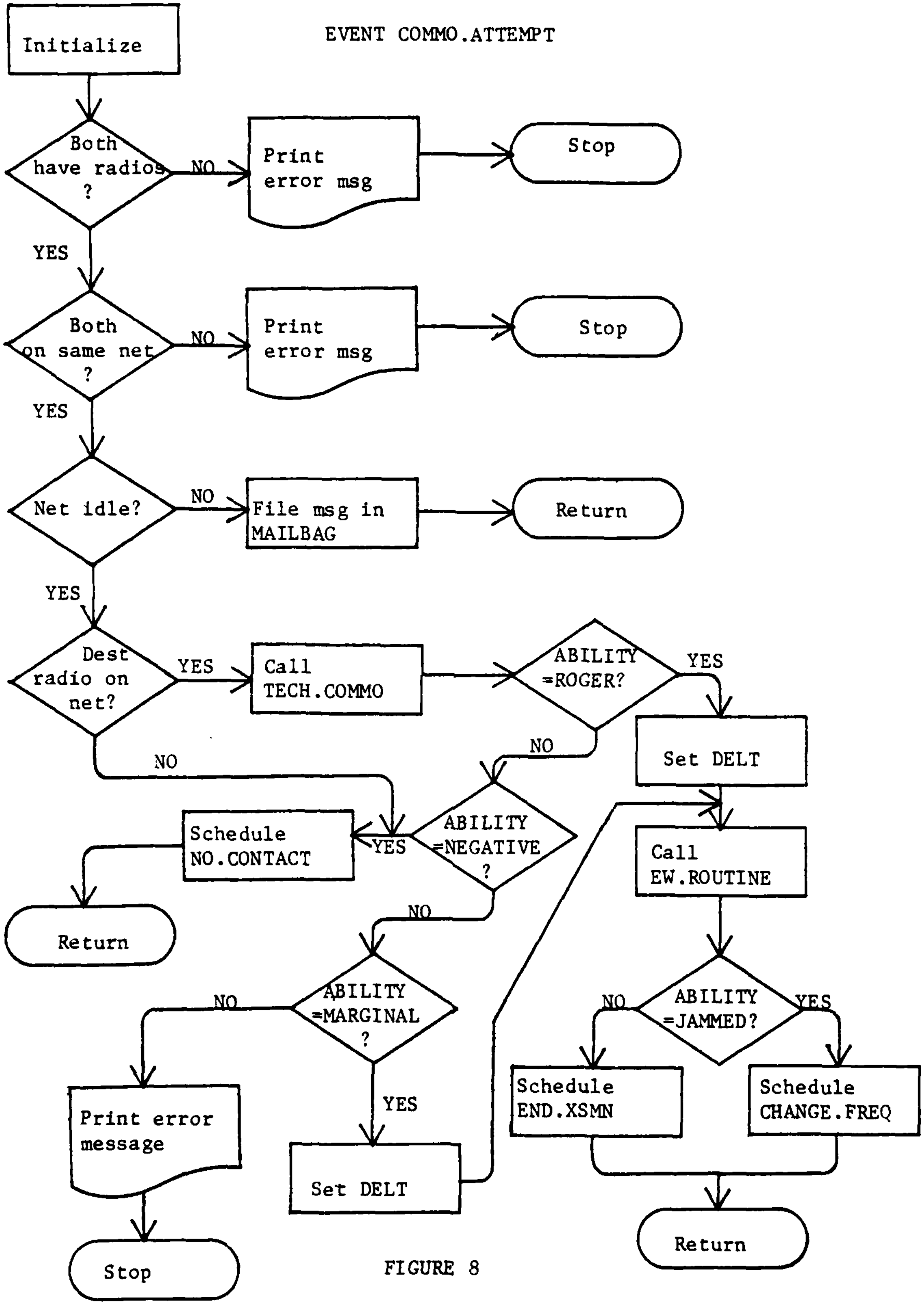

EVENT COMMO.ATTEMPT
Initialize
Both have radios ?
NO
Print error msg
Stop
YES
Both on same net ?
NO
Print error msg
Stop
YES
Net idle?
NO
File msg in MAILBAG
Return
YES
Dest radio on net?
YES
Call TECH.COMMO
ABILITY =ROGER?
YES
Set DELT
NO
NO
ABILITY =NEGATIVE ?
YES
Schedule NO.CONTACT
Return
Call EW.ROUTINE
NO
ABILITY =MARGINAL ?
NO
YES
Print error message
Stop
Set DELT
ABILITY =JAMMED?
NO
YES
Schedule END.XSMN
Schedule CHANGE.FREQ
Return

FIGURE 8

Appendix Y. Routine TECH.COMMO determines whether communication between two specific radios is technically possible. It returns one of four values for the ABILITY of the communication. It is either ROGER, meaning good, or MARGINAL, or JAMMED, or NEGATIVE meaning there was no answer by the distant station. If the ABILITY is JAMMED, event CHANGE.FREQ is scheduled and the program returns from event COMMO.ATTEMPT. If the ABILITY is NEGATIVE, event NO.CONTACT is scheduled and the program returns from event COMMO.ATTEMPT. If the ABILITY is either ROGER or MARGINAL, routine EW.ROUTINE is called to determine if the messaqge will be jammed during transmission. If the EW.ROUTINE call indicates that the transmission is jammed, event CHANGE.FREQ is scheduled and the program returns from event COMMO.ATTEMPT. If the transmission is not jammed, event END.XSMN is scheduled and the program returns from event COMMO.ATTEMPT. It currently generates the four possible results by doing a random draw against the four integers 0, 1, 2, and 3. An initial procedure has been written to generate these values for ABILITY by a detailed algorithm based upon the radio characteristics, the terrain, and the locations of the two SUBSCRIBERs. This detailed algorithm will be enhanced and eventually it will replace the random draw currently in use.

N. EVENT CHANGE.FREQ

The SIMSCRIPT code for event CHANGE.FREQ is provided in Appendix Z. Event CHANGE.FREQ simulates the delay caused by having to change the net frequency as a result of its being jammed. It sets the attributes of the net and the calling radio to idle and then reschedules the COMMO.ATTEMPT. Bedause the scheduling of the CHANGE.FREQevent includes the

delay time that would be caused by changing the frequency, the COMMO.ATTEMPT is rescheduled "now." Also, because the probability of jamming is independent and there is no memory of the frequency, the frequency is not actually changed.

O. ROUTINE EW.ROUTINE

The SIMSCRIPT code for routine EW.ROUTINE is provided in Appendix AA. Routine EW.ROUTINE simulates the possibility of a transmission being detected and jammed by enemy action. At present, this is a random draw from 0, 1, 2, and 3 with equal probabilities. A draft module to model the direction finding and jamming activities explicitly has been written and it will be enhanced and will eventually replace the current uniform random draw.

P. EVENT NO.CONTACT

The SIMSCRIPT code for event NO.CONTACT is provided in Appendix BB. Event NO.CONTACT simulates the occurance of no response by the distant station. It sets the status of the net and the calling radio to idle and then calls routine SIEZE.NET before filing the current MESSAGE in the MAILBAG of the calling radio.

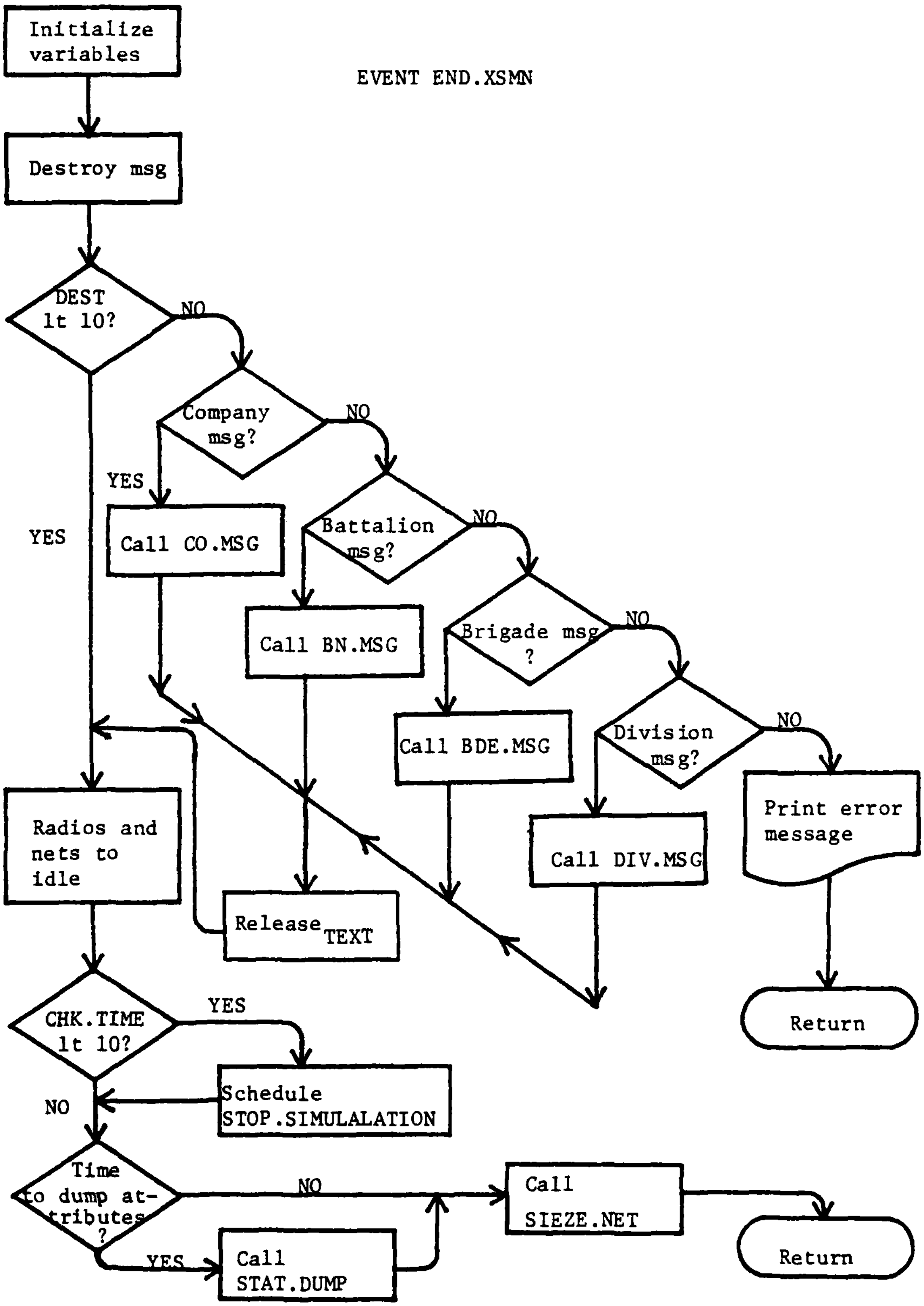
Q. ROUTINE SIEZE.NET

The SIMSCRIPT code for routine SIEZE.NET is provided in Appendix CC. Routine SIEZE.NET does a random draw against all the SUBSCRIBERs on a specific FMNET which have MESSAGESs waiting in their MAILBAGs to determine which of them will

get the net which has been left idle as a result of the occurance of either a NO.CONTACT or END.XSMN event. It then removes that MESSAGE from its MAILBAG and schedules a COMMO.ATTEMPT "now."

R. EVENT END.XSMN

The SIMSCRIPT code for event END.XSMN is provided in Appendix DD. The flow diagram for the logic of event END.XSMN is provided in Figure 9. Event END.XSMN simulates the arrival of the MESSAGE at the distant station. It gathers the message identification data into global variables and then destroys the MESSAGE. It then tests the value of the the destination code of the message to determine which level of the logic should handle the MESSAGE. This is possible since every MESSAGE has both a type and a destination code and because the text of the message where the data for the decision process is stored is set up as a separate variable length record for which only a pointer is stored in the actual MESSAGE. This means that each type message can have a different length text, and the contents can be generated separately from the actual fixed format portion of the MESSAGE. The attributes of each MESSAGE are passed as arguments through the communications process. A diagram of the pointer links used in the communications logic is provided in Figure 2. Based upon the destination code, one of the message distribution routines, CO.MSG, BN.MSG, BDE.MSG, or DIV.MSG is called. It then releases the text where the MESSAGE data was stored and sets the status of the calling radio, the receiving radio, and the FMNET to idle. It then checks the simulation time using the FORTRAN routine MYTIME to see if the allocated CPU time for the execution of the program is about to run out.



EVENT END.XSMN
Initialize variables
Destroy msg
DEST lt 10?
NO
YES
Company msg?
NO
YES
Call CO.MSG
Battalion msg?
NO
Call BN.MSG
Brigade msg ?
NO
Call BDE.MSG
Division msg?
NO
Call DIV.MSG
Print error message
Return
Radios and nets to idle
Release TEXT
CHK.TIME lt 10?
YES
Schedule STOP.SIMULALATION
NO
Time to dump attributes ?
NO
YES
Call STAT.DUMP
Call SIEZE.NET
Return

FIGURE 9

If less than 10 seconds of time remain, a STOP.SIMULATION is scheduled "now." Regardless of the time remaining, the time since the last printout of the attributes of the commanders is checked and if it is has been approximately (within 10%) COM.DATA.DUMP seconds since the last print, routine STAT.DUMP is called. Finally, routine SIEZE.NET is called and the program returns from event END.XSMN.

S. ROUTINES CO.MSG, BN.MSG, BDE.MSG, and DIV.MSG

The SIMSCRIPT code for all of these routines is provided in Appendix EE. Routines CO.MSG, BN.MSG, BDE.MSG, and DIV.MSG all act as distribution activities which test the value of the type of the MESSAGE and call the appropriate logic routine to handle that MESSAGE. In future versions of the model there will be many different types of messages corresponding to the different functions to be accomplished by the messages such as calling for air support, artillery fires, or logistics support. Routines DEC.CO, DEC.BN, DEC.BDE, and DEC.DIV are all called from these .MSG routines.

T. ROUTINE DEC.CO

The SIMSCRIPT code for routine DEC.CO is provided in Appendix I. In addition, a flow diagram of the logic for routine DEC.CO is provided in Figure 4. Routine DEC.CO handles the movement decision logic at the company level. It is called from routine CO.MSG based upon the value of the MSG.TYPE attribute of the MESSAGE. It first stores the text of the MESSAGE in global variables and then tests for the specific type of the MESSAGE. The MESSAGE types are

delineated in the PREAMBLE as global "DEFINE TO MEAN" statements. Based upon the value of the MSG.TYPE, several different actions are taken. If the MESSAGE restricts the company from moving, the company's permission attribute is set to zero, a line noting that the company has been ordered not to move is printed and the program returns. If the MESSAGE gives the company permission to move, the company's permission attribute is set to one and a line noting that the company has been given permission to move is printed before further processing of the MESSAGE is attempted. If the MESSSAGE orders the company to move, the company's permission attribute is set to one, a line noting that the company has been ordered to move is printed, and then movement of each TANK in the company is initiated. Routine AIR.COMMO and routine CO.GO are called for each TANK and a RE.MV.STATE is scheduled for the company in 60 seconds. At this point, the value of ORDER is set to two and routine EXIT is called prior to the return from the DEC.CO routine. If the message falls past the above described procedures which provide returns, it then causes the location of the company to be determined. Then it checks to see if the company has been given permission to move. If it has, ORDER is set to one. If not, ORDER is not reset from the value it has on entry to this section of code. Then routine EXIT is called and the program returns.

U. ROUTINE DEC.BN

The SIMSCRIPT code for routine DEC.BN is provided in Appendix J. A flow diagram of the logic for routine DEC.BN is provided in Figure 5. This routine handles the movement decision logic at the battalion level. It is called from routine BN.MSG based upon the value of the MSG.TYPE

attribute of the MESSAGE being processed. It stores the text of the MESSAGE in global variables and then tests for the specific type of the MESSAGE. If the MESSAGE is a request to move from a company which has not previously requested permission to move, the value of COWT is added to BNCUR and the request count of the company is set to one before any furthur processing is attempted. It then checks to see if the MESSAGE is a notification of a company or section move. If it is a company movement notification, a line noting that fact is printed and then a MESSAGE is generated and sent to inform the brigade that a company has moved. If it is a section movement notification, a line noting that fact is printed and the program returns. If the MESSAGE is an order to the battalion to move, the battalion's permission attribute is set to one and the BNGO value is added toe the BNCUR value prior to further processing. If the MESSAGE is permission for the battalion to move, the battalion's permission attribute is set to one and the BNCUR value is set equal to the BNLO threshold. If the MESSAGE orders the battalion not to move, the battalion's permission attribute is set to zero and the program returns without any further processing. When a MESSAGE reaches this point, the originator and addressee are interchanged to facilitate subsequent responses. At this point, the permission status of the battalion is checked. If it is a one, the battalion has permission to move and subsequent tests are done to see if the BNCUR value has exceeded the BNLO or the BNGO thresholds. First, the BNGO bound is checked and if it has been exceeded, a MESSAGE is generated and sent to each company in the battalion ordering the companies to move. Also, a line is printed stating that the companies have been ordered to move and a MESSAGE is generated and sent informing the brigade that the companies have all been ordered to move. If the BNLO threshold has

been exceeded, each company is sent a MESSAGE which gives it permission to move and a line is printed noting that fact. Also, a MESSAGE is sent to the brigade with the information that all the companies have been given permission to move. If the BNCUR value doesn't exceed BNLO, the company is sent a MESSAGE ordering it not to move and a line is printed noting this action. If the battalion is restricted from moving and the BNCUR value exceeds the BNLO bound, the battalion sends a MESSAGE to the brigade requesting permission to move. Finally, if the battalion is restricted and the BNCUR value does not exceed the BNLO bound, the battalion sends a MESSAGE to the originating company ordering it not to move. Then the program returns from routine DEC.BN.

V. ROUTINE DEC.BDE

The SIMSCRIPT code for routine DEC.BDE is provided in Appendix K. A flow diagram of the logic for routine DEC.BDE is provided in Figure 6. This routine handles the movement decision logic at the brigade level. It is called from routine BDE.MSG based upon the value of the MSG.TYPE of the MESSAGE being processed. The logic of this routine is exactly symmetric to the logic in routine DEC.BN with battalion, brigade, and division substituted for company, battalion, and brigade respectively. The attributes used must be those for the brigade vice those for the battalion.

W. ROUTINE DEC.DIV

The SIMSCRIPT code for routine DEC.DIV is provided in Appendix L. A flow diagram for the logic of routine DEC.DIV

is provided in Figure 7. This routine handles the movement decision logic at the division level. It is called from routine DIV.MSG based upon the value of the MSG.TYPE of the MESSAGE being processed. The logic of this routine is the same as that at the battalion and brigade level except that it is somewhat truncated since no messages are sent to a higher command. The attributes used must be those for the division level and brigade and division must be substituted for battalion and brigade.

X. EVENT DUMP.MAILBAG

The SIMSCRIPT code for event DUMP.MAILBAG is provided in Appendix FF. Event DUMP.MAILBAG simulates the event of going through all the MAIL.BAGs of all the SUBSCRIBERs and checking to see if the time of the simulation has exceeded the LST.SEND.TIME for any of the MESSAGEs in any of the MAILBAGs. Each MESSAGE which is too old is removed from its MAILBAG and routine ABORT.MSG is called for it. Event DUMP.MAILBAG then reschedules itself in 600 seconds.

Y. ROUTINE ABORT.MSG

The SIMSCRIPT code for routine ABORT.MSG is provided in Appendix GG. Routine ABORT.MSG simulates the action of sending a MESSAGE by another means if it gets to be a certain age rather than to continue to wait for the transmission. Currently, it simply destroys the MESSAGE, but, it is planned to upgrade this routine so that it will simulate the transmission by courier or some other means explicitly.

Z. EVENT MSG.GEN

The SIMSCRIPT code for event MSG.GEN is provided in Appendix HH. Event MSG.GEN simulates the background noise messages which monopolize the communications nets and with which the movement decision messages must compete for time on the nets. It generates messages at random on all the simulated nets. It first checks to insure there are at least two SUBSCRIBERs on the net. Then it selects at random one of these SUBSCRIBERs to be the originator of the MESSAGE and another of them to be the destination of the MESSAGE. It then creates the MESSAGE, sets its attributes, increments the NXT.MSG.NO for the originating station, and schedules the COMMO.ATTEMPT to send the MESSAGE. It then selects at random the next FMNET on which a "noise" MESSAGE will be sent and reschedules itself to generate the MESSAGE on that net.

VII. COMMUNICATIONS MODULE PARAMETERIZATION

There are several features of the communications parameterization process which warrant detailed examination and they are covered in the paragraphs below.

The communications between any two receivers in the same net depends on several variables. One of these variables is the ability of the radios to send a signal of sufficient strength so that it is greater than the squelch threshhold of the distant receiver. This is a function of radio line-of-sight, polarity of the antennas, power output of the radios, and many other factors. In the present implementation of communications, this capability is modeled as a uniform random draw against an integer variable between 0 and 3. An ability of 0 means the communication is feasible. An ability of 1 means the communication is marginal and requires two complete transmissions to get the message through. The effect of this is to tie up the net for twice as long as if the ability had been 0. An ability of 2 means the distant station does not answer. An ability of 3 means that successful jamming was encountered and a frequency change is scheduled prior to attempting to send the message again. Once the ability is determined by routine TECH.COMMO, there is another factor which must be considered when the message is actually being transmitted, the ability of the enemy to selectively detect and jam the communications. In the current version of the communications model, this ability is modeled as a uniform distribution with a 1/4 probability of jamming the transmission and a 3/4 probability of no jamming. In the final version of the communications model the distribution of the jamming will be a user input which can be any

suitably defined distribution function. In addition, it is anticipated that a module will be used to calculate the transmission loss using the STAR LOS (line of sight) routine, the power and transmission characteristics of the radios, and the data on the antennas. This transmission loss will be translated into a 0, 1, 2, or 3 to give a clear, a partial, a blocked or a jammed transmission respectively.

The ABORT.MSG routine is currently used to drop old messages from the lists of those waiting to be sent. In subsequent revisions of this logic, it is anticipated that an alternate method of delivery for these messages will be developed, and they will be sent by this alternate means if they get to be a certain age. Currently, they are simply destroyed if they reach the designated age without being transmitted. The age at which the destruction is invoked is a user input.

The user has two important parameters at his disposal to assist him in debugging any problems in the communications code or the input data. The first of these, COM.PRINT, determines which of the builtin messages are printed out by the program during execution. There are several levels for this parameter. It is tested for being greater than 10, 15, 20, and 25 at various points in the logic. The larger the value of this parameter used, the more detailed will be the printout of data during the execution. The second parameter, COM.DATA.DUMP, determines the amount of simulated time which is allowed to pass before a dump of the attributes of the company,battalion, brigade, and division commanders is executed. This attribute dump enables the user to check to be sure that the correct weights are being used for the units and that the request and permission attributes of the commanders are being properly set by the program. Both of these parameters are read in from cards at the beginning of the STAR BIG.MAIN routine.

VIII. COMMUNICATIONS MODULE INPUT REQUIREMENTS

The input requirements for the communications module fall into several categories based upon which module reads them in and uses them. The folowing descriptions are given in the same order as they occur when executing the model and are broken down into sections by routine. All read statements use the free form read with at least one blank separating variables.

BIG.MAIN

A. Read COM.PRINT
   COM.PRINT
      This integer variable determines the level of detail which is printed out during execution of the model. When set to zero, all of the print statements are turned off. The major break points in the printout decision logic are set at 10, 15, 20, and 25. Any COM.PRINT value larger than these levels causes all those levels below it to be turned on.

B. Read COM.DATA.DUMP
   COM.DATA.DUMP
      This real variable determines how often the values of the attributes of the unit commanders are printed out. For example, if read in as 600.0, approximately every 600 seconds of simulation time the attributes will be printed.

BUILDNETS

A. Read NONETS,NORELAY
   NONETS
      This integer variable specifies the total number of FM nets which are to be simulated.
   NORELAY
      This integer variable specifies the total number of relays to be simulated.

B. Read NFMN, NRSUB, NALTFR
   NFMN
      This integer variable specifies the number designator of the ith radio net.
   NRSUB
      This integer variable specifies the number of alternate frequencies assigned to the ith radio net.
   NALTFR
      This integer variable specifies the number of alternate

frequencies assigned to the ith radio net.

C. Read NTCK and NPRI
NTCK
This integer variable is a check to insure that the correct frequency is assigned to each net. It must be the same as NFMN from B. above to avoid an error.
NPRI
This integer variable is the primary frequency assigned to the ith net.

D. Read TKHOLD
TKHOLD
This integer variable is the number designator of the UNIT/TANK which belongs to the SUBSCRIBER created in this loop. This SUBSCRIBER will be a node in the net designated by NTCK and NFMN. All the SUBSCRIBER number designators are read in and the SUBSCRIBERs filed in the net before going on to read statement E.

E. Read NETNO
NETNO
This integer variable must equal NFMN to avoid an error.

F. Read TKALT
TKALT
This integer variable has the alternate frequencies of net NFMN read into it one at a time. All these alternate frequencies are read and stored in the ragged array FM.PUSHES in a loop prior to going to the next read statement B.

COMMGEAR

A. Read NRADIO.TANKS
NRADIO.TANKS
This integer variable specifies the number of tanks in the simulation which will have radio equipment.

B. Read TKNO, NORAD
TKNO
This integer variable specifies the number designator of the TANK/UNIT which will have communications equipment.
NORAD
This integer variable specifies the number of sets of radio equipment (RT.CONFIG's) the TANK/UNIT will have.

C. Read RAD.TYPE(RT.CONFIG) and MODE.OPERATION(RT.CONFIG)
RAD.TYPE(RT.CONFIG)
This integer attribute of RT.CONFIG specifies the type of radio in use. At present these are dummy types and have no real significance.
MODE.OPERATION(RT.CONFIG)
This integer attribute of RT. CONFIG specifies the mode of operation of the radio. At present these are dummy modes and have no real significance.

D. Read ANT.TYPE(RT.CONFIG)
ANT.TYPE(RT.CONFIG)
This integer attribute of RT. CONFIG specifies the type of antenna being used for this radio set. If this number is less than zero, the antenna is directional,

and since there are no provisions for this type in the rest of the program at this time, this causes the program to print a message concerning directional antennas and stop.

E. Read REM.ANT(RT.CONFIG)
   REM.ANT(RT.CONFIG)
   This integer attribute of RT.CONFIG specifies whether or not the antenna is remoted from the radio. If the value is not zero, the antenna is remoted, and since there is no provision for this in the rest of the program, it causes the program to print a message concerning remoted antennas and stop.

INITIALIZE

A. Read PAR00, PAR01, ... PARxx
   PARxx
   These are integer parameters used for generating random numbers using the SIMSCRIPT random number calls. The value of xx should not exceed 49, and each should be different.

B. Read PAR50, PAR51, ... PARyy
   PARyy
   These are real parameters which are used for random number generation. The yy value should be larger than 49, and each yy must be unique.

C. Read CO.LO.BD and CO.UP.BD
   CO.LO.BD
   This integer variable is the lower bound of the destination code for messages being sent to the company level.
   CO.UP.BD
   This integer variable is the upper bound of the destination code for messages being sent to the company level.

D. Read BN.LO.BD and BN.UP.BD
   BN.LO.BD
   This integer variable is the lower bound of the destination code for messages being sent to the battalion level.
   BN.UP.BD
   This integer variable is the upper bound of the destination code for messages being sent to the battalion level.

E. Read BDE.LO.BD and BDE.UP.BD
   BDE.LO.BD
   This integer variable is the lower bound of the destination code for messages being sent to the brigade level.
   BDE.UP.BD
   This integer variable is the upper bound of the destination code for messages being sent to the brigade level.

F. Read DIV.LO.BD and DIV.UP.BD
   DIV.LO.BD This integer variable is the lower bound of the destination code for messages being sent to the division level.
   DIV.UP.BD

This integer variable is the upper bound of the destination code for messages being sent to the division level.

IX. FUTURE MODEL ENHANCEMENETS

The model of communications described in this document represents a good foundation for developing those additional communications-related modules required to enhance the realism of the STAR Model. The structure of this model is sufficiently flexible to handle most of the communications requirements envisioned for inclusion in STAR.

Obviously, there is still much to be done. A few of the more important areas for future work are identified in this chapter.

Enhancement of the red interception and jamming capability module initially developed by Cpt. Haislip is high on the list of significant actions which will have a marked effect on the realism of the current model since it incorporates a call to a jamming routine for which the user supplies parameters.

In addition to the red electronic warfare capabilities, the blue force EW assets and doctrine must be evaluated and modeled to provide a balanced picture of the impact of EW on the model.

The play of relays for the FM nets, radio wire integration, radio teletype HF nets, and directional antennas should be considered for inclusion in the model. It may be that some of these do not provide sufficient gains in realism to justify their being included, but they should be studied.

Since the STAR model is headed for the goal of modeling a blue brigade under attack by a red division, the multichannel radio nets within the brigade boundary and those going back to the corps level need to be studied and considered for inclusion in the model. These nets represent

a large proportion of the communications assets in the brigade area and should be included in the model in some manner. Additionally, as more and more of the logistics and support functions are included in the model, the need to model communications to entities other than the unit commanders will increase. This means modeling at least part of the communications network down to the individual instrument or line where these support people are located.

Finally, the whole area of Command, Control, Communications, and Intelligence needs to be integrated in some way with the tactical decision logic in order to take advantage of the information which can be gained from the intelligence network.

APPENDIX A

STAR ROUTINES, EVENTS, AND ARRAYS

The following routines, events, and arrays are key elements in the existing STAR movement decision logic.

ROUTINES

Action
This routine calls the movement decision routine and takes action based on the value of the order returned to initiate movement of units in conjunction with the leave logic.

Bn.go
This routine initiates movement of a blue battalion from its current position.

Bug.chk
This routine checks to see if a unit should move as a result of being too close to an enemy unit and calls the movement decision routine if required.

Co.go
This routine initiates movement of a blue company from its current position.

Decision
This routine is called by routine ACTION or routine BUG.CHK. It updates the values of CREQST, BREQST, BNCUR, and BDECUR and checks the attributes of a unit and the values stored in the arrays TABLE, BNCORD, and COCORD to determine if the unit is allowed to perform a desired movement. It causes units to move if the BNCUR or BDECUR values exceed the LO or GO bounds for the battalion or the brigade.

EVENTS

Phaz.chk
This event determines whether the current phase line is still occupied. If it is still occupied, the event reschedules itself. If it isn't occupied, the event removes the battalion commander from the BRIGADE (set) and calls routine COORD.SET.

Re.mv.state
This event checks to see if units have arrived at their new locations.

ARRAYS

Table

A diagram of the organization of this array is provided in Figure 10. This three dimensional array has a plane for each blue maneuver unit and seven rows for each plane. The first six rows correspond to monitored weapons systems for the unit and the seventh corresponds to the unit as a whole. Each weapon system row has 4 plus 3*NPRI columns where NPRI is the number of attrition level actions planned for that system. Column 1 contains the system code in the first 6 rows and the boundary range in the seventh row. If no enemy units are within this range, the unit will not move off its position unless ordered to do so. Column 2 contains the weapon code in the first six rows and the range within which all enemy units are used for computing the force ratio in the seventh row. Column 3 contains the closing range within which an enemy firing at a system causes it to request permission to move off its position. Column 4 contains the code for whether a system or unit is restricted from moving or is free to move at will. The remaining columns are in groups of 3 with the first being an attrition level, the second being the force ratio, and the third being the code for the action to be taken if that attrition level and force ratio are exceeded.

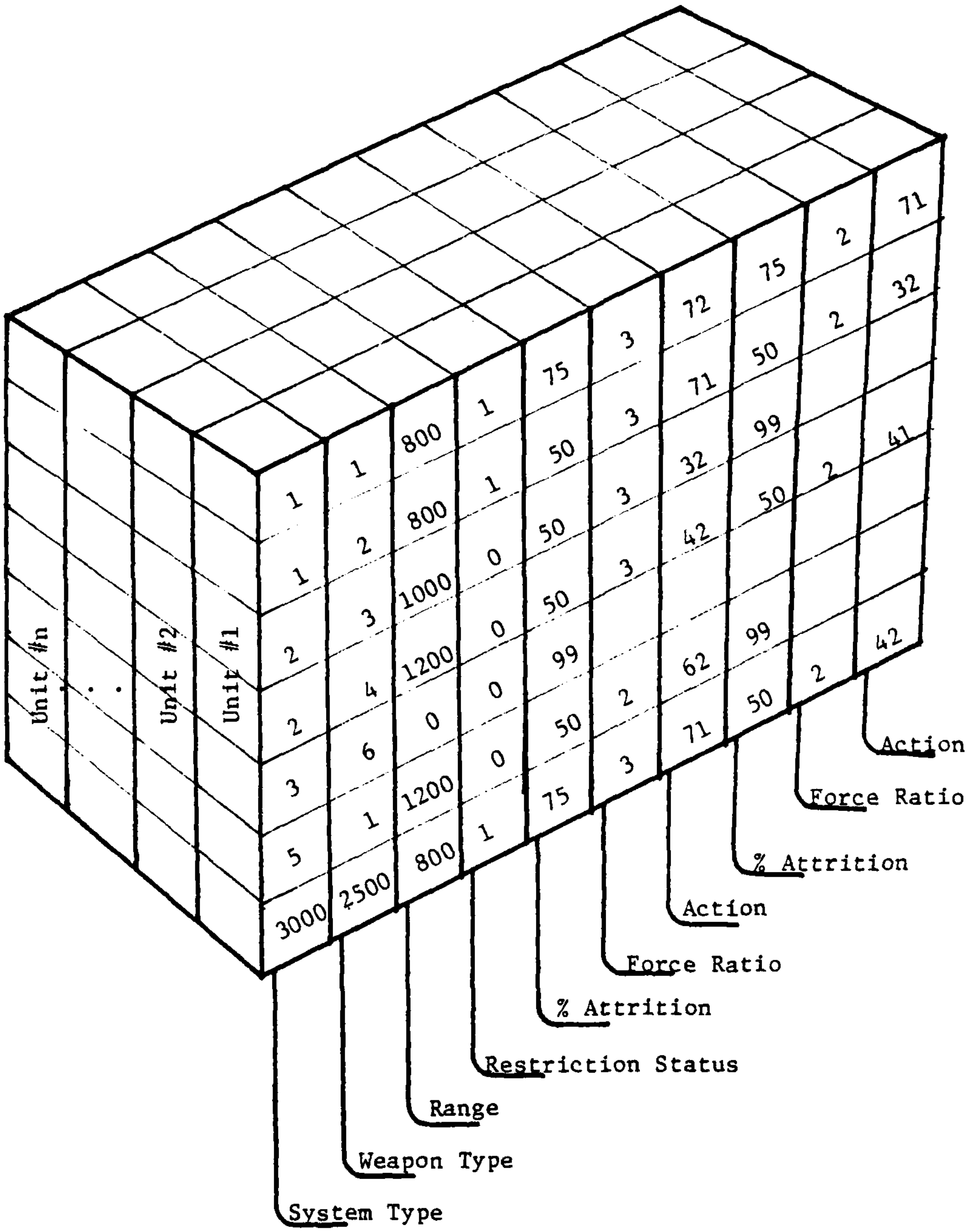
Bncord

A diagram of the organization of this array is provided in Figure 11. This three dimensional array has a plane for each blue battalion, a row for each phase line, and five columns. The first column contains BMSN, the code for whether or not the battalion has permission to move off this coordination line. The second column contains BNCUR, the sum of the weights of the companies which have requested permission to move off their current coordination lines. The third column contains BNWT, the tactical weight of the battalion on this coordination line. The fourth column contains BNLO, a lower bound which if exceeded by BNCUR causes the battalion to give all the companies permission to move. The fifth column contains BNGO, a lower bound which if exceeded by BNCUR causes the Battalion to order all companies to move.

Cocord

A diagram of this array is provided in Figure 12. This three dimensional array has a plane for each blue company unit, a row for each coordination line, and two columns, The first column contains CMSN, the code for whether or not the company has permission to move off its current coordination line. The second column contains COWT, the tactical weight of the company on this coordination line.

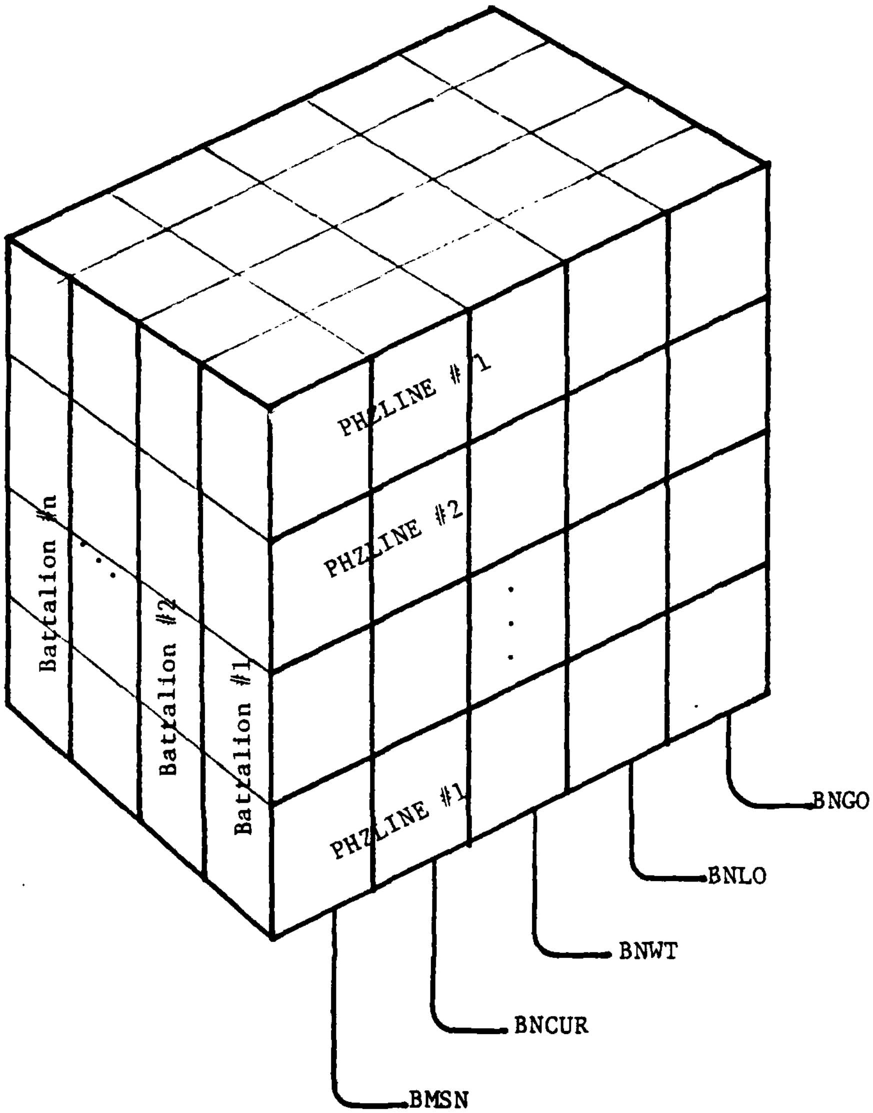
TABLE DIAGRAM



A plane for each blue maneuver unit
A column in each plane for each of the above weapon system attributes
A row in each plane for each of up to seven weapon systems

FIGURE 10

BNCORD DIAGRAM



A plane for each battalion

A row in each plane for each phase line

A column in each plane for each of the five attributes

FIGURE 11

COCORD DIAGRAM

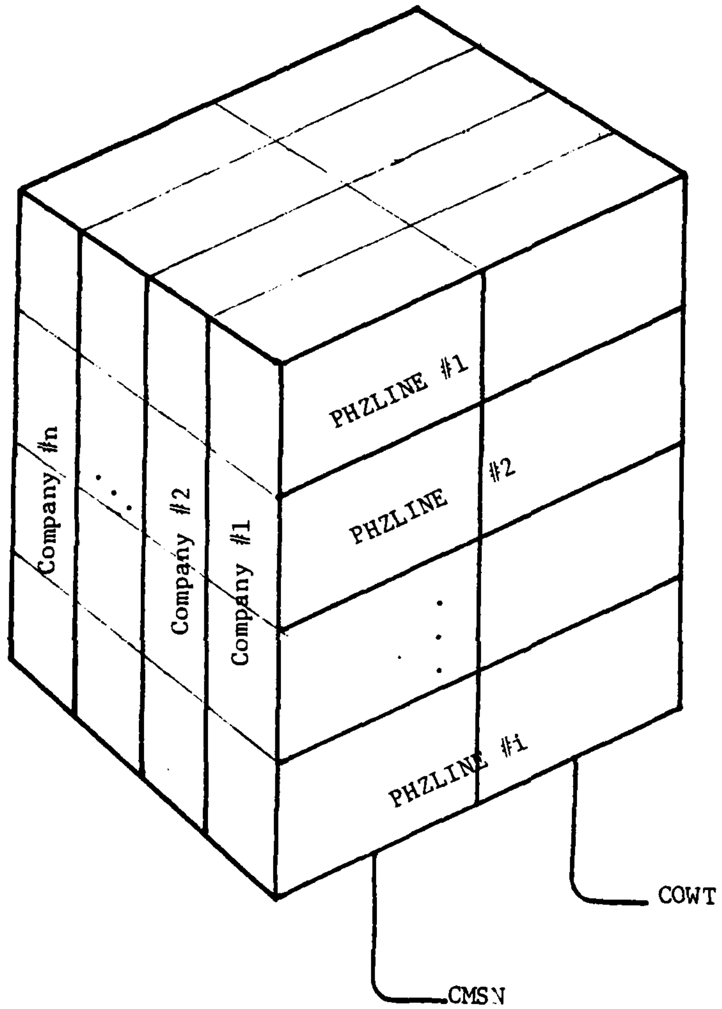


FIGURE 12

APPENDIX B

COMMUNICATIONS ROUTINES AND EVENTS

The communications model uses the following routines and events to model the transmission of messages between the decision levels. The decision process is based upon the information contained in the messages and their texts.

ROUTINES

Abort.msg(tank,msg)
This routine destroys a message if it gets too old.

Bde.msg
This routine is called by event END.XSMN whenver a brigade-level message is encountered. This routine in turn calls the appropriate logic routine to handle the message based on the message type.

Bn.msg
This routine is called by event END.XSMN whenever a battalion-level message is encountered. This routine in turn calls the appropriate logic routine to handle the message based on the message type.

Bnettbl
This routine is called by routine COM.MAIN during initialization of the program. It builds two two-dimensional tables or ragged arrays. The first, NETTBL, stores the subscripts of the subscribers of the Ith radio net in row I of the array. The second, RPOINT, stores in its Ith row the address pointers of the subscribers in the Ith radio net.

Buildnets
This routine is called by routine COM.MAIN during initialization of the program. It reads in the number of radio nets, the number of relays, the subscribers in each net, and the alternate frequencies for each net. It creates the FMNETS and files them in the ETHER and also creates the SUBSCRIBER's and files them in the CEOI.LIST of the FMNET to which they belong.

Checknets
This routine is called by routine COM.MAIN during initialization of the program. It cycles through each FMNET and prints the number of the net, the primary frequency of the net, the number of alternate frequencies, the alternate frequencies, and the number of SUBSCRIBER's in the net. This is primarily a data debugging facility.

Co.msg

This routine is called by event END.XSMN whenever a company-level message is encountered. This routine in turn calls the appropriate logic routine to handle the message based on the message type.

Com.main
This routine calls the routines which initialize the communications portion of the program. It then schedules the DUMP.MAILBAG and MSG.GEN(FMNET) events and returns control to the SIMSCRIPT II.5 scheduler.

Commcheck
This routine is called by routine COM.MAIN during initialization of the program. It cycles through every TANK in BLUE.ALIVE and prints out the communications gear which has been assigned to each TANK, the location of each TANK, the mode of operation of each radio, and the type of antenna used by each radio. This is primarily a data debugging facility.

Commgear
This routine is called by routine COM.MAIN during initialization of the program. It reads the number of each TANK which is to have communications capability and the number of radios each TANK is to have. It then creates a MYSET for that TANK which is a set containing pointers to all the RT.CONFIG's (radios) owned by that TANK. It creates the required number of RT.CONFIG's for that TANK and files them in RAD.LIST of MYSET. It reads the radio type, the mode of operation of the radio, and the antenna type used by the radio and sets these attributes of each of the RT.CONFIG's created. After creating all of the radios, the routine sets the next message number of each tank to n001, where n is the number of the TANK. It then places the equipment in the appropriate net, and prints out a matrix of the TANK subscript and the address pointers for the TANK, the MYSET, the RT.CONFIG, and the SUBSCRIBER for each TANK in each FM net in the ETHER. This listing is primarily a debugging facility.

Dec.bde
This routine handles the movement decision at the brigade level. It checks to see if a battalion should move or be given permission to move and if so, it causes the appropriate message to be generated and schedules the COMMO.ATTEMPT to the battalion.

Dec.bn
This routine handles the movement decision at the battalion level. It causes messages to be generated and schedules COMMO.ATTEMPT's to both the company level and the brigade level.

Dec.co
This routine initiates movement of the company from its current position by calling the CO.GO routine if authorized by the text of the message.

Dec.div
This routine handles the movement decision logic at the division level. It causes messages to be generated and schedules COMMO.ATTEMPT's to the brigade level.

Decision(row,col,t)
This routine does the preliminary checks to determine if a company needs permission to move and if so, it

causes a message to be generated and schedules a COMMO.ATTEMPT to the battalion. If permission is not required, it causes the unit to begin movement.

Div.msg

This routine is called by the END.XSMN event whenever a division-level message is encountered. This routine then calls the appropriate logic routine to handle the message based on the message type.

Ew.routine

This routine is called by the COMMO.ATTEMPT event whenever a message is being placed on the net to determine if the message will be jammed. If the message is jammed, event CHANGE.FREQ is scheduled and executed prior to allowing another attempt to send the message.

Exit(order,t,crit,p.z.time,flag)

This routine is used as an escape from the decision process if permission is not required for a company to move from its current position.

Gen.move.decision.msg(pharow,order,t,crit,p.z.time) yielding msg

This routine produces the movement request and order messages, assigns values to the message text variables, and passes a pointer to the message back to the calling routine.

Initialize

This routine is called by routine COM.MAIN during the initialization of the communications portion of the program. This routine reads the parameters for the distributions used in scheduling the communications events. It also calls routines SSTAT.DUMP and STAT.DUMP.

Sieze.net(callnet)

Upon termination of a message on any of the FMNET's, this routine checks that FMNET to see if there are any messages which have been saved and need to be transmitted over it. If so, it selects one at random and schedules the COMMO.ATTEMPT for that message.

Sstat.dump.

This routine is called by routine INITIALIZE during the initialization of the program. This routine prints out the values of the attributes of the brigade, battalion, and company commanders which are used for movement coordination.

Stat.dump

This routine is called by routine INITIALIZE during the initialization of the program and by routine SNAP.R if the program should fail. This routine lists the attributes of each company commander, battalion commander, brigade commander, and division commander.

Tech.commo(callrt,recrt) yielding ability

This routine is used to determine how good communication is between the origin and destination stations attempting to pass and receive each message. A message can be one of four types: 1) normal transmission, 2) marginal transmission (It takes twice as long as a normal transmission.), 3) no response (The distant station does not answer.), and 4) jammed transmission (The sender experiences jamming as soon as

he enters the net and immediately schedules a CHANGE.FREQ as a result.).

Xy.antenna(rt.config) yielding xt, yt, and zt
This routine determines the x, y, and z coordinate locations of remoted antennas. It is called by routine COMMGEAR during the initialization phase of the program.

EVENTS

Change.freq(callrt,callnet,msg)
This event is called when the attempted transmission is jammed. It "changes" the tranmitting frequency of the net. In fact, the frequency isn't changed since there is no memory property for the jamming routine and each test to see if a MESSAGE is jammed is an independent trial. The frequency could be changed by a quite simple algorithm, but since this isn't required, another COMMO.ATTEMPT is scheduled after a suitable wait representing the time required to change the frequency.

Commo.attempt(orig,msg)
This event simulates the process of acquiring communications with a distant station. It checks to insure that both UNIT's/TANK's have radios, that they are both on the same net, that communication is technically feasible, that the net is not busy, and that the distant station responds to the call. It files the MESSAGE in a MAILBAG if it can't be transmitted and schedules another COMMO.ATTEMPT at a later time. It also sets the busy-state of the FMNET to busy.

Dump.mailbag
This event checks every station in every net for messages waiting to be transmitted which have a LST.SEND.TIME attribute less than simulation time. When it finds a MESSAGE which satisfies this condition, it removes the MESSAGE from the MAILBAG of those waiting for transmission and calls ABORT.MSG which destroys the MESSAGE.

End.xsmn(callrt,recrt,callnet,msg)
This event is scheduled when the COMMO.ATTEMPT for each MESSAGE occurs. It is the current simulated time plus the length of the MESSAGE. When this event occurs, the MESSAGE content is examined and based upon the destination code, one of the DEC. routines is called to handle the decision process. This event also sets the busy-state of the FMNET to idle.

Msg.gen(fmnet)
This event generates "background noise" messages on the nets to simulate the other traffic with which the movement decision messages have to compete for time on the net.

No.contact(callrt,callnet,msg)
This event simulates a lack of response by the distant station by setting the busy-state of the net to idle so that other messages can be passed. It also files the MESSAGE in a MAILBAG for future transmission.

APPENDIX C

COMMUNICATIONS GLOBAL VARIABLES

INTEGER VARIABLES

Addr.store
: This is a temporary storage location for the address to which a MESSAGE will be sent. It is used to pass the address to the message generation routine rather than increase the number of parameters explicitly passed to the routine.

Adrsee
: This is a temporary location used in the decision logic to store the addressee of an incoming MESSAGE. It is loaded by the END.XSMN event just prior to the destruction of the MESSAGE.

Bbde
: This variable is one that was added to those used in the basic STAR model. It contains the value read in for the number of blue brigades to be played.

Bde.lo.bd
: This is an input parameter set by the user as the lower bound of the DEST.CODE of a MESSAGE that will be sent to the brigade.

Bde.up.bd
: This is an input parameter set by the user as the upper bound of the DEST.CODE of a MESSAGE which will be sent to the brigade.

Bn.lo.bd
: Same as BDE.LO.BD but for the battalion level.

Bn.up.bd
: Same as BDE.UP.BD but for the battalion level.

Co.lo.bd
: Same as BDE.LO.BD but for the company level.

Co.up.bd
: Same as BDE.UP.BD but for the company level.

Com.print
: This input parameter determines what information is printed during an execution of the model. The larger COM.PRINT is, the more detailed the printout becomes. When set to zero, the execution printout is minimized. Current break points for increasing the printed output are set at 10,15,20, and 25. Any value larger than any of these will cause that level of print to be activated.

Dest
: This is a temporary location used to store and test the destination level of a MESSAGE. The value is set in event END.XSMN just prior to destruction of the

MESSAGE. Subsequent to the MESSAGE's destruction, DEST is tested to determine which command level must take action on the MESSAGE.

Div.lo.bd
Same as BDE.LO.BD but for the division level.

Div.up.bd
Same as BDE.UP.BD but for the division level.

Dvn
This variable was added to those in use by the STAR model. The number of divisions to be simulated is read into it during initialization of the model.

Flag
This is a temporary storage location used by the print routines. It contains an integer value which uniquely identifies the exit which was taken from the movement decision logic. It is used primarily as an aid to debugging.

Msg
This is the core location of the current MESSAGE. It is a pointer which uniquely identifies the MESSAGE.

Msgno
This is a temporary storage location for the number which is used to distinguish one message between two entities from another. This variable is set by the END.XSMN event just prior to destruction of the MESSAGE.

Orig
This is a temporary storage location for the pointer to the originating UNIT of a MESSAGE. It is set in routine END.XSMN just prior to destruction of the MESSAGE.

Parxx
These are input parameters which are used in the calls to various distributions for the purpose of generating random numbers which are used to schedule events. In general, if xx is between 00 and 49, this should be an integer number.

Text.pointer
This variable is a temporary location used to store the pointer to the TEXT which is associated with a particular MESSAGE. It is set in routine END.XSMN just prior to destruction of the MESSAGE.

Type
This is a temporary location used to store the value of the message type. The value is set in routine END.XSMN just prior to destruction of the MESSAGE, and it is tested in the BN.MSG, BDE.MSG, CO.MSG, and DIV.MSG routines after destruction of the MESSAGE to determine which of the logic routines to call to handle the MESSAGE.

REAL VARIABLES

Chk.time
This is a variable used to store the value returned

from the Fortran function MYTIME in event END.XSMN. The MYTIME routine returns the number of ten thousandths of a second left before the program uses up its allotted CPU time. This number is divided by 10000 and truncated before being stored in CHK.TIME. If the value stored in CHK.TIME is less than 10, a STOP.SIMULATION is scheduled immediately so that the data up to that point will be printed out and not lost as it would be if the job crashed for exceeding its allocated time.

Com.data.dump
: This is an input parameter which determines how often the STAT.DUMP routine will be executed to print out the attributes of each CO.COMMANDER, BN.COMMANDER, BDE.COMMANDER, and DIV.COMMANDER. For example, if COM.DATA.DUMP is read in as 600.0 by the user, approximately every 600 seconds of simulation time the attributes will be printed out.

Com.time
: This variable is used to store the value of TIME.V each time the routine STAT.DUMP is called. It is used to insure that STAT.DUMP is only called once every COM.DATA.DUMP seconds of simulation time.

Paryy
: These are input variables used as parameters for the distributions provided by the Simscript language for scheduling events and generating random numbers. If yy is a value from 50 to 99, the parameter should be a real variable.

Rni
: This variable is used in routine SIEZE.NET to select at random the MESSAGE which will get the net from all those stored in MAILBAGs which are assigned to the net.

APPENDIX D

COMMUNICATIONS PERMANENT ENTITIES

BDE.COMMANDER

Bdecur

This attribute is the sum of the weights of all the battalions in the BRIGADE which have requested permission to move from the current coordination line. It is reset to zero each time the brigade first occupies a coordination line.

Bdego

This attribute is a user input upper bound. If BDECUR equals or exceeds this value, all the battalions in the BRIGADE are ordered to move back to their next coordination line position. A distinct BDEGO value is linked to each coordination line which the brigade can occupy.

Bdelo

This attribute is a user input lower bound. If BDECUR equals or exceeds this value, all the battalions in the BRIGADE are given permission to move at will back to their next coordination line position. A distinct BDELO value is linked to each coordination line which the brigade can occupy.

Bdemsn

This user input attribute is stored in the three-dimensional array, BDECORD. It indicates whether or not the brigade has permission to move from a given coordination line. A 1 means that the brigade has permission to move and a 0 means that the brigade does not have permission to move.

Bdereqst

This attribute indicates whether or not a brigade has requested permission to move from its current coordination line poosition. A 1 means that the brigade has requested permission and a 0 means that the brigade has not requested permission to move.

Bdewt

This user input attribute indicates the tactical weight or relative importance of the brigade's position on a particular coordination line. There is a distinct BDEWT value for each of the coordination lines which the brigade can occupy.

Brde

This attribute stores the number designator of the brigade.

No.bde.unit

This attribute stores the number designator of the UNIT/TANK which is associated with the brigade commander.

BN.COMMANDER

Batt
This attribute stores the number designator of the battalion.

Bmsn
This user input attribute indicates whether or not the battalion has permission to withdraw from a particular coordination line. This attribute is stored in the three-dimensional array, BNCORD. A 1 means that the battalion has permission to move while a 0 means that the battalion does not have permission to move.

Bncur
This attribute stores the sum of the weights of the companies which have requested permission to move from their current coordination line position. It is reset to zero each time the battalion first occupies a coordination line.

Bngo
This user input attribute is an upper bound. If BNCUR equals or exceeds this value, all the companies in the BATTALION will be ordered to move back to their next coordination line position. A distinct BNGO value is linked to each coordination line which the battalion can occupy.

Bnlo
This user input attribute is a lower bound. If BNCUR equals or exceeds this value, each company in the BATTALION is given permission to move at will back to its next coordination line position. A distinct BNLO value is linked to each coordination line which the battalion can occupy.

Bnwt
This user input attribute indicates the tactical weight or relative importance of the battalion's position on a given coordination line. There is a distinct BNWT value for each coordination line which the battalion can occupy.

Breqst
This attribute indicates whether or not the battalion has requested permission to move from its current coordination line position. A 1 means that the battalion has requested permission to move while a 0 means that the battalion has not requested permission to move.

No.bn.unit
This attribute stores the number designator of the UNIT/TANK which is associated with the battalion commander.

CO.COMMANDER

Cmsn

This user input attribute stored in the three-dimensional array, COCORD, indicates whether a company has permission to move from a given coordination line position. A 1 means the company has permission to move while a 0 means that the company does not have permission to move.

Compy

This attribute stores the number designator of the UNIT/TANK which is associated with the company commander.

Cowt

This user input attribute indicates the tactical weight or relative importance of the company's position on a given coordination line. There is a distinct COWT value for each coordination line which the company can occupy.

Creqst

This attribute indicates whether the company has requested permission to move from its current coordination line position. A 1 means that the company has requested permission to move while a 0 means that the company has not requested permission to move.

DIV.COMMANDER

Ddiv

This attribute stores the number designator of the division.

Divcur

This attribute stores the sum of the weights of the brigades which have requested permission to move from their current coordination line position. It is reset to zero each time the division first occupies a coordination line.

Divgo

This attribute is a user input upper bound. If DIVCUR equals or exceeds this value, all the brigades in the DIVISION will be ordered to move back to their next coordination line position. A distinct DIVGO value is associated with each coordination line which the division can occupy.

Divlo

This user input attribute is a lower bound. If DIVCUR equals or exceeds this value, each brigade in the DIVISION is given permission to move at will back to its next coordination line position. A distinct DIVLO value is associated with each coordination line which the division can occupy.

Divreqst

This attribute indicates whether the division has requested permission to move from its current coordination line position. A 1 means that it has requested permission to move and a 0 means that it has not requested permission to move.

Divwt

This attribute is a user input which indicates the tactical weight or relative importance of the division's position on a given coordination line. There is a distinct DIVWT for each coordination line which the division can occupy.

No.div.unit

This attribute stores the number designator of the UNIT/TANK which is associated with the division commander.

B.MVR.CDR

Prty

This attribute is used to determine the correct plane of the TABLE array which is to be used for a given UNIT on a given coordination line.

There are many other attributes of the B.MVR.CDRs, but this is the only one which has any importance in the communications model. The rest of the attributes are used for combat analysis of casualties and related statistics.

APPENDIX E

COMMUNICATIONS TEMPORARY ENTITIES

FMNET

Fm.idle
This attribute indicates whether or not the net is in use, with a code of 1, or idle, with a code of 0.

Fmalt.freq
This user input attribute contains the alternate frequency of the FMNET.

Fmpri.freq
This attribute contains the user input primary frequency of the FMNET.

Ncs.point
This attribute contains a pointer to the TANK/UNIT of the net control station for the FMNET.

Num.net
This attribute indicates the numeric designator of this FMNET.

MESSAGE

Addressee
This attribute stores the pointer to the UNIT/TANK which is associated with the commander who is the addressee of the message.

Dest.code
This attribute stores a code value which indicates the level to which the message is being sent. For example, if the value of this variable lies between the user input values for CO.LO.BD and CO.UP.BD, the message will be sent to the company level.

Length.msg
This attribute stores the length of the message in simulated seconds. It is used during event COMMO.ATTEMPT to determine when to schedule the END.XSMN event.

Lst.send.time
This attribute stores the simulated time beyond which the message will not be transmitted. It is used in the DUMP.MAILBAG event to determine if the message should be aborted.

Msg.no
This attribute stores the number of the message in the form nxxx where n is the number of the UNIT/TANK sending the message and xxx is a sequential integer which starts at one and increases one for each message sent by this UNIT/TANK.

Msg.text

This attribute stores the pointer to the text which is created to go with this message.

Msg.type

This attribute indicates the type of the message. For example, a type 30 message is a request for permission to move from the battalion to the brigade.

MYSET

Nxt.msg.no

This attribute stores the number designator of the next message which this particular TANK/UNIT will send.

RT.CONFIG

Ant.type

This attribute contains a code which describes the type of antenna which is used by the radio configuration. If the value is less than zero, the antenna is directional. If the value is greater than zero, the antenna is omnidirectional.

Ant.x.loc

This attribute contains the x coordinate of the location of the antenna which may not be the same as the location of the TANK/UNIT which uses that antenna because remote antennas are allowed.

Ant.y.loc

This attribute contains the y coordinate of the location of the antenna.

In.use

This attribute contains a code which is set to 1 to indicate that the radio set is in use and is set to 0 when the set is not being used.

Mode.operation

This attribute contains a code which designates the mode of operation of the radio.

Own.subscriber

This attribute contains a pointer to the subscriber which owns the TANK/UNIT to which this RT.CONFIG belongs.

Rad.type

This attribute contains a code which designates the type of radio.

Rem.ant

This attribute contains a code which tells if the antenna for this RT.CONFIG is remoted or not. Any value other than zero indicates a remoted antenna.

SUBSCRIBER

Own.net

This attribute contains a pointer to the FMNET which is used.

Own.tank
This attribute contains a pointer to the UNIT/TANK associated with this subscriber.

Tfc.waiting
This attribute contains a code of 1 if there is at least one message filed in this SUBSCRIBER's mailbag waiting to be sent. Otherwise, the value of this attribute is 0.

APPENDIX F

COMMUNICATIONS ARRAYS

BDECORD

The BDECORD array stores five different attributes of the brigades. It is a three dimensional array which has a plane for each blue brigade. Within each plane there is a row for each phase line which the brigade can occupy and within each row there are five columns. One column for each of the five attributes which will be stored. The first column contains BDEMSN, the code for whether or not the brigade has permission to move off this coordination line. The second column contains BDECUR, which is the sum of the weights of the battalions which have requested permission to move from their coordination line positions. The third column contains BDEWT, which is the user supplied estimate of the tactical importance of the brigade on this coordination line. The fourth column contains BDELO, the lower bound which if exceeded by BDECUR causes the brigade to give all its battalions permission to move from their current coordination line positions. The fifth column contains BDEGO, the upper bound which if exceeded by BDECUR causes the brigade to order all its battalions to move from their current coordination line positions. This array is used to reset the attributes of the brigade commanders after they first occupy a coordination line position. The array itself, with the exception of BDEMSN and BDECUR values is loaded during the initialization of the program using input data supplied by the user. A diagram of the way this array is organized is provided in Figure 13.

BNCORD

This array is the same one that is used by the production STAR model. It is explained in Appendix A. A diagram of the organization of the array is provided in Figure 11.

BNETTBL

This two dimensional array is used by the communications routines as a table lookup which contains the subscript number of the UNITS/TANKS which belong to the FMNETs. The subscripts of the TANKS which belong to the ith FMNET are elements of the ith row of the array. Because the FMNETs are not restricted to all having the same number of SUBSCRIBERs, this array is necessarily ragged. An example of this array is provided in Figure 14.

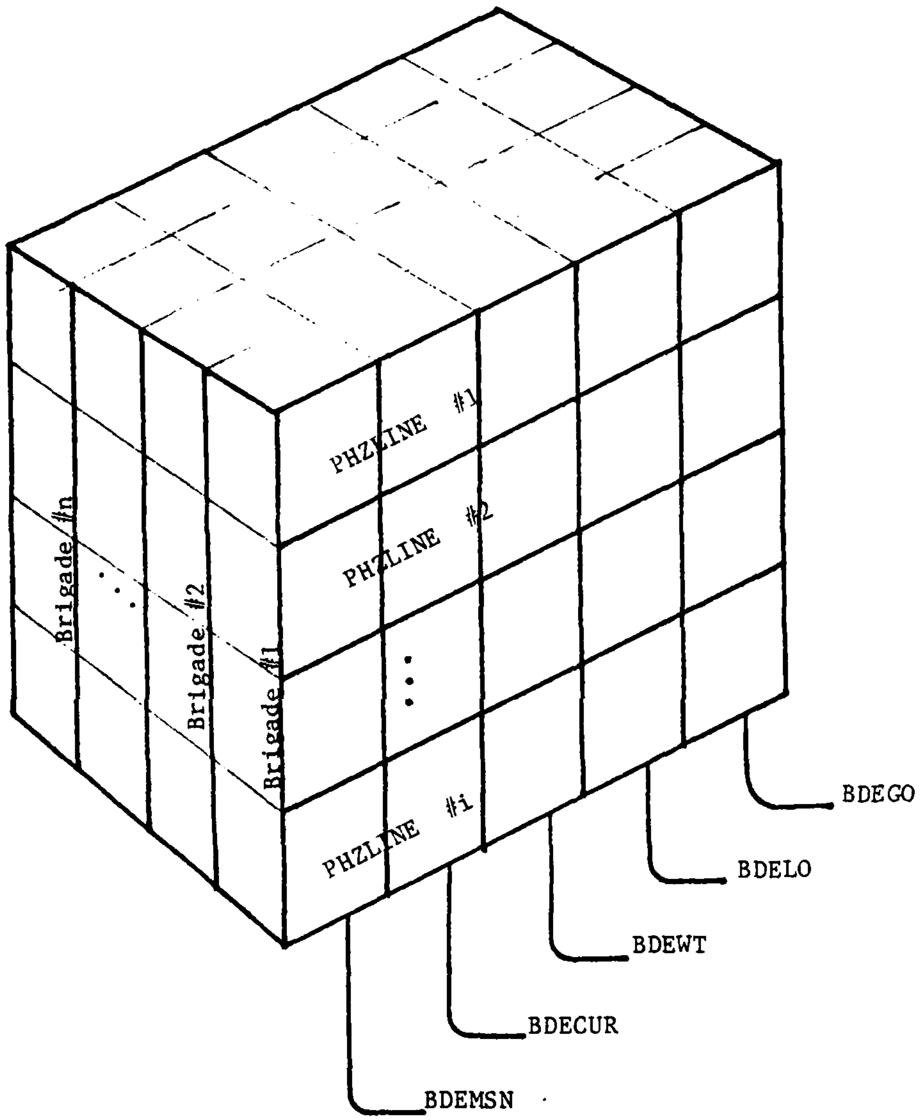
COCORD

This array is the same one that is used by the production STAR model. It is explained in Appendix A. A diagram of the organization of the array is provided in Figure 12.

DIVCORD

This array is similar to the BNCORD AND BDECORD arrays. It contains the same type of data but for the division level as opposed to the brigade level. A diagram of the way this three dimensional array is

BDECORD DIAGRAM



A plane for each brigade

A row in each plane for each phase line

A column in each plane for each of the five attributes

FIGURE 13

BNETTBL DIAGRAM

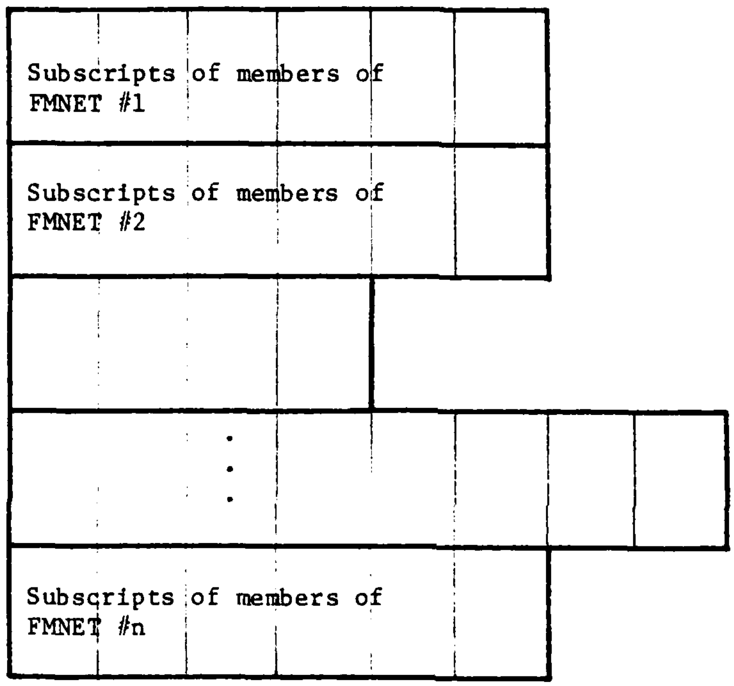


FIGURE 14

organized is provided in Figure 15.

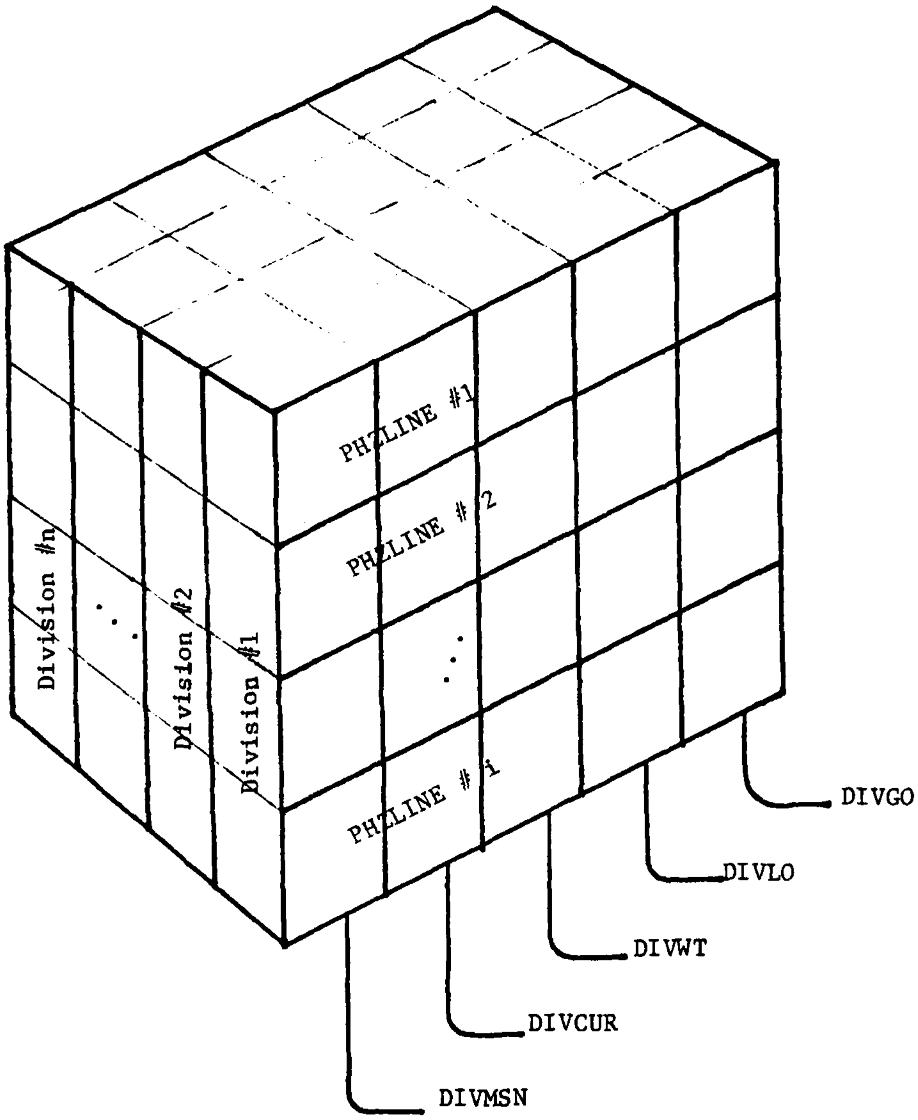
RRPOINT

This two dimensional array is used primarilly as an aid to debugging. It contains the pointer values of the UNITs/TANKs which belong to the ith radio net in the ith row of the array. This array is released after the initialization of the program where it is printed out since the pointers never change during the simulation. A diagram of the organization of this array is provided in Figure 16.

TABLE

This three dimensional array is the same as the one used by the production version of STAR. It is explained in Appendix A. A diagram of the organization of the array is provided in Figure 10.

DIVCORD DIAGRAM



A plane for each division

A row in each plane for each phase line

A column in each plane for each of the five attributes

FIGURE 15

RRPOINT DIAGRAM

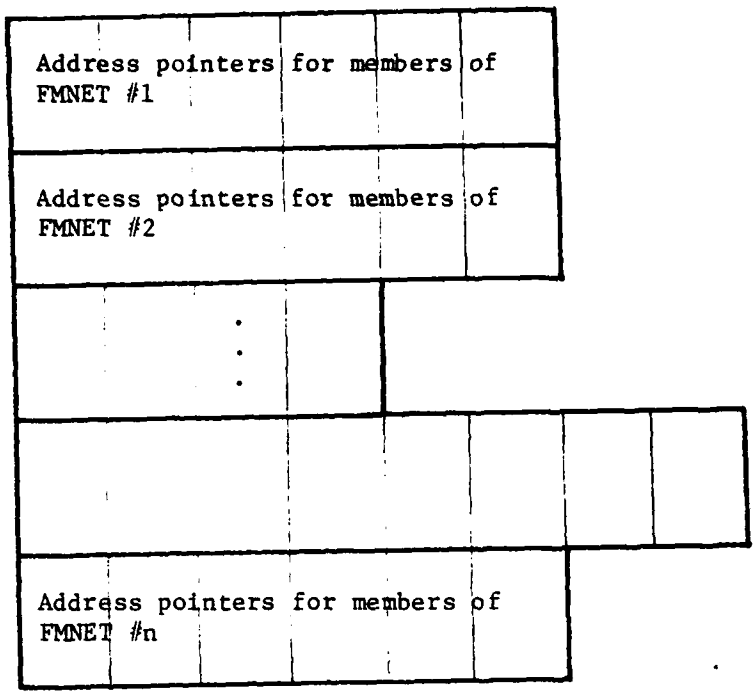


FIGURE 16

APPENDIX G

COMMUNICATIONS SETS

Ceoi.list
This set contains pointers to all the SUBSCRIBERs who have access to a specific FMNET.

Ether
This set contains all the FMNET's which are simulated by the communications model.

Mailbag
This set contains pointers to all the MESSAGEs generated by a given SUBSCRIBER for a given FMNET which haven't been sent.

Rad.list
This set contains pointers to all the RT.CONFIGS of a given TANK.

Figure 2 contains a complete diagram of all the sets and pointers used by the communications model.

APPENDIX H

```
ROUTINE DECISION(ROW,COL,T)

          ''THIS ROUTINE EXAMINES THE STATUS OF THE SECTION OR COMPANY TO WHICH
              ''TANK T BELONGS AND DETERMINES WHETHER OR NOT MOVEMENT IS ALLOWED

IF COM.PRINT GT 20 PRINT 1 LINE WITH TIME.V THUS
     INTO DECISION AT *****.***
ALWAYS
LET ORDER= NO
IF MVARY(ROW,COL) GE 4 LET ORDER=YES
     IF COM.PRINT GT 15
     .    PRINT 1 LINE WITH ROW AND COL AS FOLLOWS
99999-----STATUS OF MVARY(ROW=*****,COL=*****) CAUSED EXIT WITH ORDER=YES
          SKIP 1 LINE
     ALWAYS
     GO TO ACT4, ACT5, ACT6 PER (MVARY(ROW,COL) - 3)
     'ACT4'
          CALL BN.GO(T,SYS,WPN)
          GO TO ACT.NOW
     'ACT5'
          CALL OTHER.GO(T,SYS,WPN)
          GO TO ACT.NOW
     'ACT6'
          CALL MOUNTER(T,SYS,WPN)
     'ACT.NOW'
          SCHEDULE A RE.MV.STATE(CO(T)) IN 60 UNITS
     RETURN
ELSE
IF TABLE(PRTY(CO(T)),ROW,4)=0

          ''THE SECTION OR COMPANY IS NOT RESTRICTED FROM MOVING AT WILL.

     LET ORDER=YES LET FLAG=1
     CALL EXIT(ORDER,T,CRIT,P.Z.TIME,FLAG)
     LET ORIG=ARRPOINT(CO(T))
     LET ADDR.STORE = ARRPOINT(NO.BN.UNIT(BN(T)))
     CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME) YIELDING MSG

          ''GENERATE A MESSAGE INFORMING THE BATTALION OF THE CO/SEC MOVE.

     LET DEST.CODE(MSG) = BN.LEVEL
     IF ROW EQ 7 LET MSG.TYPE(MSG) = CO.TELLS.BN.OF.CO.MOVE
               CALL CO.GO(T,SYS,WPN)
               SCHEDULE A RE.MV.STATE(CO(T)) IN 60 UNITS
          IF RANGE(CHK.RNG) LE TABLE(PRTY(CO(T)),ROW,3)
               PRINT 1 LINE AS FOLLOWS
     +++ DECISION BASED ON RANGE TO ENEMY ++++++
          ALWAYS
     ELSE
          LET MSG.TYPE(MSG) = CO.NOTIF.BN.OF.SEC.MOVE
```

```
                IF RANGE(CHK.RNG) LE TABLE(PRTY(CO(T)),ROW,3)
                     PRINT 1 LINE AS FOLLOWS
           *** DECISION BASED ON RANGE TO ENEMY ******
                     CALL VEH.GO(T,TABLE(PRTY(CO(T)),I,1),TABLE(PRTY(CO(T)),I,2))
                     CALL CHG.STATUS(T)
                     RETURN
                ELSE
           LET SYS=TABLE(PRTY(CO(T)),ROW,1)
           LET WPN=TABLE(PRTY(CO(T)),ROW,2)
           CALL CHG.STATUS(T)
           GO TO ACT1, ACT2 PER MVARY(ROW,COL)
           'ACT1'
                CALL SEC.GO(T,SYS,WPN)
                GO TO ACT.LATER
           'ACT2'
                CALL PLT.GO(T,SYS,WPN)
           'ACT.LATER'
                SCHEDULE A RE.MV.STATE(CO(T)) IN 60 UNITS
           ALWAYS
           SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) NOW
           RETURN
      ELSE
      IF CREQST(CO(T)) EQ 1
                ''THIS COMPANY HAS PREVIOUSLY REQUESTED PERMISSION TO MOVE.
           IF COM.PRINT GT 15
                PRINT 1 LINE AS FOLLOWS
      11111-----CREQST(CO(T)) EQ 1 CAUSED A RETURN
           ALWAYS
           RETURN
      ELSE
      LET ACHK=TRUNC.F(AREA.START(T)/100)
      IF ACHK GT PHAROW
                ''DETERMINE IF THE COMPANY IS ON THE SAME PHASE LINE AS THE REST OF
                     ''ITS SISTER UNITS.  IF IT IS FARTHER TO THE REAR, DO NOT ALLOW A
                     ''REQUEST TO MOVE.
           IF COM.PRINT GT 15
                PRINT 1 LINE WITH ACHK AND PHAROW AS FOLLOWS
      22222-----STATUS OF ACHK=***  GT PHAROW=*** CAUSED A RETURN
           ALWAYS
           RETURN
      ELSE
      IF COCORD(CO(T),ACHK,1)=1
                ''THE COMPANY IS RESTRICTED BUT HAS BEEN GIVEN PERMISSION TO MOVE.
           LET ORDER=YES LET FLAG=2
```

```
    CALL EXIT(ORDER,T.CRIT,P.Z.TIME,FLAG)
    LET ORIG=RRRPOINT(CO(T))
    LET ADDR.STORE = RRRPOINT(NO.BN.UNIT(BN(T)))
    CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T.CRIT,P.Z.TIME) YIELDING MSG
        ''GENERATE A MESSAGE INFORMING THE BATTALION OF THE CO/SEC MOVE.
    LET DEST.CODE(MSG) = BN.LEVEL
    IF ROW EQ 7 LET MSG.TYPE(MSG) = CO.TELLS.BN.OF.CO.MOVE
            CALL CO.GO(T,SYS,WPN)
            SCHEDULE A RE.MV.STATE(CO(T)) IN 60 UNITS
        IF RANGE(CHK.RNG) LE TABLE(PRTY(CO(T)),ROW,3)
            PRINT 1 LINE AS FOLLOWS
    +++ DECISION BASED ON RANGE TO ENEMY ++++++
        ALWAYS
    ELSE
        LET MSG.TYPE(MSG) = CO.NOTIF.BN.OF.SEC.MOVE
        IF RANGE(CHK.RNG) LE TABLE(PRTY(CO(T)),ROW,3)
            PRINT 1 LINE AS FOLLOWS
    +++ DECISION BASED ON RANGE TO ENEMY ++++++
            CALL VEH.GO(T,TABLE(PRTY(CO(T)),I,1),TABLE(PRTY(CO(T)),I,2))
            CALL CHG.STATUS(T)
            RETURN
        ELSE
    LET SYS=TABLE(PRTY(CO(T)),ROW,1)
    LET WPN=TABLE(PRTY(CO(T)),ROW,2)
    CALL CHG.STATUS(T)
    GO TO BCT1, BCT2 PER MVARY(ROW,COL)
    'BCT1'
        CALL SEC.GO(T,SYS,WPN)
        GO TO BCT.LATER
    'BCT2'
        CALL PLT.GO(T,SYS,WPN)
    'BCT.LATER'
        SCHEDULE A RE.MV.STATE(CO(T)) IN 60 UNITS
    ALWAYS
    SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) NOW
    RETURN
ELSE
LET FLAG=3
LET ORIG=RRRPOINT(CO(T))
LET ADDR.STORE = RRRPOINT(NO.BN.UNIT(BN(T)))
CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME) YIELDING MSG
        ''GENERATE A MESSAGE TO THE BATTALION REQUESTING PERMISSION TO MOVE.
LET DEST.CODE(MSG) = BN.LEVEL
LET MSG.TYPE(MSG) = CO.BN.REQ.PERM.TO.MOVE
SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) NOW
RETURN
```

```
      END
```

APPENDIX I

```
ROUTINE DEC.CO
DEFINE J AS AN INTEGER VARIABLE

IF COM.PRINT GT 20 PRINT 1 LINE WITH TIME.V THUS
     ENTERED DEC.CO ROUTINE AT TIME.V=*****.***
ALWAYS
LET TEXT(*) = TEXT.POINTER
LET PHAROW = TEXT(1)
LET ORDER = TEXT(2)
LET T = TEXT(3)
LET CRIT = TEXT(4)
LET P.Z.TIME = TEXT(5)
LET TEXT(*)=0
IF CO.CAN.NOT.MOVE
     LET COCORD(CO(T),PHAROW,1) = 0
     IF COM.PRINT GT 15
          PRINT 1 LINE WITH CO(T) THUS
     COMPANY ** HAS BEEN ORDERED NOT TO MOVE
     ALWAYS
     SKIP 2 LINES
     RETURN
ELSE
IF CO.MAY.MOVE
     IF COM.PRINT GT 15
          PRINT 1 LINE WITH CO(T) THUS
     COMPANY ** HAS BEEN GIVEN PERMISSION TO MOVE AT WILL
          SKIP 2 LINES
     ALWAYS
     LET COCORD(CO(T),PHAROW,1) = 1
ALWAYS
IF CO.ORDERED.TO.MOVE
     IF COM.PRINT GT 15
          PRINT 1 LINE WITH CO(T) THUS
     COMPANY ** HAS BEEN ORDERED TO MOVE
          SKIP 2 LINES
     ALWAYS
     LET COCORD(CO(T),PHAROW,1) = 1
     FOR EACH TANK IN BLUE.ALIVE WITH CO(TANK)= COMPY(COMPANY.COMMANDER)
          UNTIL J=1, DO
          CALL AIR.COMMO(TANK,3)  LET CRIT=4
          CALL CO.GO(TANK,0,0)
          SCHEDULE A RE.MV.STATE(CO(TANK)) IN 60 UNITS
          LET J=J+1
     LOOP
     LET ORDER=2
     LET FLAG = 4
     CALL EXIT(ORDER,T,CRIT,P.Z TIME,FLAG)
     LET FLAG = 0
     RETURN
ELSE
```

```
LET ACHK = TRUNC.F(AREA.START(T)/100)
IF COCORD(CO(T),ACHK,1)=1
     LET ORDER=YES
ALWAYS
LET FLAG = 15
CALL EXIT(ORDER,T,CRIT,P.Z.TIME,FLAG)
LET FLAG = 0
RETURN
END
```

APPENDIX J

```
ROUTINE DEC.BN

IF COM.PRINT GT 20 PRINT 1 LINE WITH TIME.V THUS
     ENTERED DEC.BN ROUTINE AT TIME.V=*****.***
ALWAYS
LET TEXT(*) = TEXT.POINTER
LET PHAROW = TEXT(1)
LET ORDER = TEXT(2)
LET T = TEXT(3)
LET CRIT = TEXT(4)
LET P.Z.TIME = TEXT(5)
LET TEXT(*)=0
IF CREQST(CO(T)) EQ 0 AND TYPE EQ CO.BN.REQ.PERM.TO.MOVE
     LET CRIT=1 LET BNCUR(BN(T))=BNCUR(BN(T)) + COWT(CO(T))
     LET CREQST(CO(T))=1
ALWAYS
IF CO.HAS.MOVED
     IF COM.PRINT GT 15
          PRINT 1 DOUBLE LINE WITH ORIG, ADRSEE, AND TEXT.POINTER AS FOLLOWS
44444+++++CO ******** NOTIFIED BN ******** BY MESSAGE (TEXT=********) THAT IT IS
 MOVING FROM ITS CURRENT POSITION.
     ALWAYS
     LET BNCUR(BN(T)) = BNCUR(BN(T)) + COWT(CO(T))
     LET FLAG=16
     LET T=RRRPOINT(NO.BN.UNIT(BN(T)))
     LET ADDR.STORE =   RRRPOINT(NO.BDE.UNIT(BDE(T)))
     CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME) YIELDING MSG
     LET DEST.CODE(MSG) = BDE.LEVEL
     LET MSG.TYPE(MSG) = BN.NOTIF.BDE.OF.CO.MOVE
     LET ORIG = T
     SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) IN COMO6.DISTR(PAR60,PAR61,4) UNITS
     RETURN
ELSE
IF SEC.HAS.MOVED
     IF COM.PRINT GT 15
          PRINT 1 DOUBLE LINE WITH ORIG, ADRSEE, AND TEXT.POINTER THUS
66666+++++CO ******** NOTIFIED BN ******** BY MESSAGE (TEXT=********) THAT ONE O
F ITS SECTIONS IS MOVING FROM ITS POSITION.
     ALWAYS
     RETURN
ELSE
IF BN.ORDERED.TO.MOVE
     LET BNCORD(BATT(BN(T)),PHAROW,1) = 1
     LET BNCUR(BN(T))=BNCUR(BN(T))+ BNGO(BN(T))
ELSE
     IF BN.MAY.MOVE
          LET BNCORD(BATT(BN(T)),PHAROW,1) = 1
          LET BNCUR(BN(T)) = BNLO(BN(T))
     ELSE
          IF BN.CAN.NOT.MOVE
```

```
                    LET BNCORD(BATT(BN(T)),PHAROW,1) = 0
                    RETURN
                ELSE
            ALWAYS
        ALWAYS
        LET TEMP=ADRSEE LET ADRSEE=ORIG LET ORIG=TEMP
        IF BNCORD(BATT(BN(T)),PHAROW,1) =1

                ''THIS BATTALION HAS PERMISSION TO MOVE AT WILL.

            IF BNCUR(BN(T)) GE BNGO(BN(T)) .

                ''THE CRITICAL STATUS OF THE BATTALION HAS EXCEEDED ITS GO THRESHHOLD.

                IF COM.PRINT GT 15
                    PRINT 1 LINE WITH BN(T), TIME.V AS FOLLOWS
            BN(**) NOT RESTRICTED AND ORDERED COMPANIES TO MOVE AT *****.***
                ALWAYS
                FOR EACH COMPANY.COMMANDER IN BATTALION(BN(T)), DO
                    LET FLAG=9
                    LET T=ARRPOINT(COMPY(COMPANY.COMMANDER))
                    LET ADDR.STORE = T
                    CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME)
                        YIELDING MSG
                    LET DEST.CODE(MSG) = CO.LEVEL
                    LET MSG.TYPE(MSG)=CO.TOLD.TO.MOVE.BY.BN
                    SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) IN
                        COMO5.DISTR(PAR58,PAR59,5) UNITS
                LOOP
                LET FLAG=15
                LET T=ARRPOINT(NO.BN.UNIT(BN(T)))
                LET ADDR.STORE = ARRPOINT(NO.BDE.UNIT(BDE(T)))
                CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME) YIELDING MSG

                ''GENERATE A MESSAGE INFORMING BDE THAT ALL COS ARE BEING ORDERED TO
                    ''MOVE AS SOON AS POSSIBLE.

                LET DEST.CODE(MSG) = BDE.LEVEL
                LET MSG.TYPE(MSG)= BN.TELLS.BDE.OF.BN.MOVE
                IF COM.PRINT GT 15
                    PRINT 1 DOUBLE LINE WITH ORIG, ADDR.STORE, AND MSG.NO(MSG) THUS
        77777+++++BN ******** NOTIFIED BDE ******** BY MESSAGE NO ****** THAT IT HAS ORD
        ERED ALL COMPANIES TO MOVE AS SOON AS POSSIBLE.
                ALWAYS
                SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) IN COMO6.DISTR(PAR60,PAR61,4) UNITS
                RETURN
            ELSE
            IF BNCUR(BN(T)) GE BNLO(BN(T))

                ''THE BATTALION CRITICAL STATUS HAS EXCEEDED THE LO THRESHHOLD.
```

```
                IF COM.PRINT GT 15
                    PRINT 1 LINE WITH BN(T), TIME.V AS FOLLOWS
            BN(**) NOT RESTRICTED AND GAVE COS PERMISSION TO MOVE AT *****.***
                ALWAYS
                FOR EACH COMPANY.COMMANDER IN BATTALION(BN(T)), DO
                    LET FLAG=10
                    LET T=ARAPOINT(COMPY(COMPANY.COMMANDER))
                    LET ADDR.STORE = T
                    CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME)
                        YIELDING MSG
                    LET DEST.CODE(MSG) = CO.LEVEL
                    LET MSG.TYPE(MSG) = BN.GAVE.CO.PERM.TO.MOVE
                    SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) IN
                        COMO5.DISTR(PAR58,PAR59,5) UNITS
                LOOP
                LET FLAG=11
                LET T=ARAPOINT(NO.BN.UNIT(BN(T)))
                LET ADDR.STORE = ARAPOINT(NO.BDE.UNIT(BDE(T)))
                CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME) YIELDING MSG

                ''GENERATE A MESSAGE INFORMING BDE THAT ALL COS HAVE BEEN GIVEN
                    ''PERMISSION TO MOVE AT WILL.

                LET DEST.CODE(MSG) = BDE.LEVEL
                LET MSG.TYPE(MSG)= BN.TELLS.BDE.OF.BN.MOVE
                IF COM.PRINT GT 15
                    PRINT 1 DOUBLE LINE WITH ORIG, ADDR.STORE, AND MSG.NO(MSG) THUS
        88888+++++BN ******** NOTIFIED BDE ******** BY MESSAGE NO ****** THAT IT HAS GIV
        EN ALL COMPANIES PERMISSION TO MOVE AT WILL.
                ALWAYS
                SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) IN COMO6.DISTR(PAR60,PAR61,4) UNITS
                RETURN
            ELSE
            IF COM.PRINT GT 15
                PRINT 1 LINE WITH BN(T), TIME.V AS FOLLOWS
            BN(**) NOT RESTRICTED AND ORDERED COMPANY NOT TO MOVE AT *****.***
            ALWAYS
            LET FLAG = 12
            LET ADDR.STORE = ADRSEE
            LET T = ADRSEE
            CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME) YIELDING MSG
            LET DEST.CODE(MSG) = CO.LEVEL
            LET MSG.TYPE(MSG)=BN.DIR.CO.NOT.TO.MOVE
            SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) IN COMO5.DISTR(PAR58,PAR59,5) UNITS
            RETURN
        ELSE
        IF BNCUR(BN(T)) GE BNLO(BN(T))
            IF COM.PRINT GT 15
                PRINT 1 LINE WITH BN(T) AND TIME.V THUS
```

AD-A094 598 NAVAL POSTGRADUATE SCHOOL MONTEREY CA F/G 17/2
PARAMETRIC SIMULATION OF TACTICAL SINGLE CHANNEL FREQUENCY MODU--ETC(U
SEP 80 P A OLSON

UNCLASSIFIED NL

2 OF 2
AD
A094 598

END
DATE
FILMED
2 -81
DTIC

```
          BN(**) IS RESTRICTED AND IS REQUESTING PERMISSION TO MOVE AT *****.***
          ALWAYS
          LET FLAG = 13
          LET T=ARRPOINT(NO.BN.UNIT(BN(T)))
          LET ADDR.STORE =   ARRPOINT(NO.BDE.UNIT(BDE(T)))
          CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME) YIELDING MSG
          LET DEST.CODE(MSG) = BDE.LEVEL
          LET MSG.TYPE(MSG)=BN.BDE.REQ.PERM.TO.MOVE
          SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) IN COMO6.DISTR(PAR60,PAR61,4) UNITS
          RETURN
     ELSE
          IF COM.PRINT GT 15
               PRINT 1 LINE WITH BN(T) AND TIME.V THUS
          BN(**) IS RESTRICTED AND ORDERED THE REQUESTING CO NOT TO MOVE AT *****.***
          ALWAYS
     LET FLAG=14
     CALL EXIT(ORDER,T,CRIT,P.Z.TIME,FLAG)
     LET ADDR.STORE = ADRSEE
     LET T = ADRSEE
     CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME) YIELDING MSG
     LET DEST.CODE(MSG) = CO.LEVEL
     LET MSG.TYPE(MSG)=BN.DIR.CO.NOT.TO.MOVE
     SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) IN COMO5.DISTR(PAR58,PAR59,5) UNITS
     RETURN
     END
```

APPENDIX K

```
ROUTINE DEC.BDE

IF COM.PRINT GT 20 PRINT 1 LINE WITH TIME.V THUS
     ENTERED DEC.BDE ROUTINE AT TIME.V=*****.***
ALWAYS
LET TEXT(*) = TEXT.POINTER
LET PHAROW = TEXT(1)
LET ORDER = TEXT(2)
LET T = TEXT(3)
LET CRIT = TEXT(4)
LET P.Z.TIME = TEXT(5)
LET TEXT(*)=0
IF BREQST(BN(T))=0 AND TYPE EQ BN.BDE.REQ.PERM.TO.MOVE
     LET CRIT=2
     LET BDECUR(BDE(T))=BDECUR(BDE(T))+BNWT(BN(T))
     LET BREQST(BN(T))=1
ALWAYS
IF BN.HAS.MOVED
     IF COM.PRINT GT 15
          PRINT 1 DOUBLE LINE WITH ORIG, ADDRESSEE(MSG), AND TEXT.POINTER THUS
55555+++++BN ******** NOTIFIED BRIGADE ******** BY MESSAGE ******** THAT IT HAS
MOVED FROM ITS POSITION
     ALWAYS
     LET BDECUR(BDE(T)) = BDECUR(BDE(T)) + BNWT(BN(T))
     LET FLAG=21
     LET T=RRRPOINT(NO.BDE.UNIT(BDE(T)))
     LET ADDR.STORE =   RRRPOINT(NO.DIV.UNIT(DIV(T)))
   . CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME) YIELDING MSG
     LET DEST.CODE(MSG) = DIV.LEVEL
     LET MSG.TYPE(MSG) = BDE.NOTIF.DIV.OF.BN.MOVE
     LET ORIG = T
     SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) IN COMO3.DISTR(PAR54,PAR55,9) UNITS
     RETURN
ELSE
IF CO.HAS.MOVED
     IF COM.PRINT GT 15
          PRINT 1 DOUBLE LINE WITH ORIG, ADRSEE, AND TEXT.POINTER AS FOLLOWS
AAAAA+++++BN ******** NOTIFIED BDE ******** BY MESSAGE (TEXT=********) THAT ONE
OF ITS COMPANIES IS MOVING FROM ITS POSITION.
     ALWAYS
     RETURN
ELSE
IF BDE.ORDERED.TO.MOVE
     LET BDECORD(BRDE(BDE(T)),PHAROW,1) = 1
     LET BDECUR(BDE(T)) = BDECUR(BDE(T)) + BDEGO(BDE(T))
ELSE
     IF BDE.MAY.MOVE
          LET BDECORD(BRDE(BDE(T)),PHAROW,1) = 1
          LET BDECUR(BDE(T)) = BDELO(BDE(T))
     ELSE
```

```
                IF BDE.CAN.NOT.MOVE
                     LET BDECORD(BRDE(BDE(T)),PHAROW,1) = 0
                     RETURN
                ELSE
           ALWAYS
      ALWAYS
      LET TEMP=ADRSEE LET ADRSEE=ORIG LET ORIG=TEMP
      IF BDECORD(BRDE(BDE(T)),PHAROW,1) = 1

                ''THIS BRIGADE HAS PERMISSION TO MOVE AT WILL.

      IF BDECUR(BDE(T)) GE BDEGO(BDE(T))

                ''THE CRITICAL STATUS OF THE BRIGADE HAS EXCEEDED ITS GO THRESHHOLD.

           LET CRIT = 4
           IF COM.PRINT GT 15
                PRINT 1 LINE WITH BDE(T) AND TIME.V THUS
           BDE(**) NOT RESTRICTED AND ORDERED BATTALIONS TO MOVE AT *****.***
           ALWAYS
           FOR EACH BN.COMMANDER IN BRIGADE(BDE(T)),DO
                LET FLAG=5
                LET T=ARRPOINT(NO.BN.UNIT(BN.COMMANDER))
                LET  ADDR.STORE = T
                CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME) YIELDING MSG
                LET DEST.CODE(MSG) = BN.LEVEL
                LET MSG.TYPE(MSG)=BN.TOLD.TO.MOVE.BY.BDE
                SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) IN COMO4.DISTR(PAR56,PAR57,7) UNITS
           LOOP
                LET FLAG = 17
                LET T = ARRPOINT(NO.BDE.UNIT(BDE(T)))
                LET ADDR.STORE = ARRPOINT(NO.DIV.UNIT(BDE(T)))
                CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME) YIELDING MSG

                ''GENERATE A MESSAGE INFORMING DIV THAT ALL BNS ARE BEING ORDERED TO
                     ''MOVE AS SOON AS POSSIBLE.

                LET DEST.CODE(MSG) = DIV.LEVEL
                LET MSG.TYPE(MSG) = BDE.LETS.DIV.KNOW.OF.BDE.MOVE
                IF COM.PRINT GT 15
                     PRINT 1 DOUBLE LINE WITH ORIG, ADDR.STORE, AND MSG.NO(MSG) THUS
      BBBBB+++++BDE ******** NOTIFIED DIV ******** BY MESSAGE NO ****** THAT IT HAS OR
      DERED ALL BATTALIONS TO MOVE AS SOON AS POSSIBLE.
                ALWAYS
                SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) IN COMO3.DISTR(PAR54,PAR55,9) UNITS
           RETURN
      ELSE
      IF BDECUR(BDE(T)) GE BDELO(BDE(T))

                ''THE BRIGADE CRITICAL STATUS HAS EXCEEDED THE LO THRESHHOLD.
```

```
    LET CRIT=3
    IF COM.PRINT GT 15
        PRINT 1 LINE WITH BDE(T) AND TIME.V THUS
    BDE(**) NOT RESTRICTED AND GAVE BNS PERMISSION TO MOVE AT *****.***
    ALWAYS
    FOR EACH BN.COMMANDER IN BRIGADE(BDE(T)), DO
        LET FLAG=6
        LET T=AAAPOINT(NO.BN.UNIT(BN.COMMANDER))
        LET ADDR.STORE = T
        CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME) YIELDING MSG
        LET DEST.CODE(MSG) = BN.LEVEL
        LET MSG.TYPE(MSG)=BDE.GAVE.BN.PERM.TO.MOVE
        SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) IN COMO4.DISTR(PARS6,PARS7,7) UNITS
    LOOP
        LET FLAG = 18
        LET T = AAAPOINT(NO.BDE.UNIT(BDE(T)))
        LET ADDR.STORE = AAAPOINT(NO.DIV.UNIT(BDE(T)))
        CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME) YIELDING MSG

        ''GENERATE A MESSAGE INFORMING DIV THAT ALL BNS HAVE BEEN GIVEN
            ''PERMISSION TO MOVE AT WILL.

        LET DEST.CODE(MSG) = DIV.LEVEL
        LET MSG.TYPE(MSG) = BDE.LETS.DIV.KNOW.OF.BDE.MOVE
        IF COM.PRINT GT 15
            PRINT 1 DOUBLE LINE WITH ORIG, ADDR.STORE, AND MSG.NO(MSG) THUS
CCCCC+++++BDE ******** NOTIFIED DIV ******** BY MESSAGE NO ****** THAT IT HAS GI
VEN ALL BATTALIONS PERMISSION TO MOVE AT WILL.
        ALWAYS
        SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) IN COMO3.DISTR(PARS4,PARS5,9) UNITS
    RETURN
ELSE
    IF COM.PRINT GT 15
        PRINT 1 LINE WITH BDE(T) AND TIME.V AS FOLLOWS
    BDE(**) NOT RESTRICTED AND ORDERED THE BATTALION NOT TO MOVE AT *****.***
    ALWAYS
    LET FLAG=7
    LET ADDR.STORE = ADRSEE
    LET T = ADRSEE
    CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME) YIELDING MSG
    LET DEST.CODE(MSG) = BN.LEVEL
    LET MSG.TYPE(MSG)=BDE.DIR.BN.NOT.TO.MOVE
        SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) IN COMO4.DISTR(PARS6,PARS7,7) UNITS
    RETURN
ELSE
IF BDECUR(BDE(T)) GE BDELO(BDE(T))
    IF COM.PRINT GT 15
        PRINT 1 LINE WITH BDE(T) AND TIME.V THUS
    BDE(**) IS RESTRICTED AND IS REQUESTING PERMISSION TO MOVE AT *****.*****
```

```
          ALWAYS
          LET FLAG = 19
          LET T=ARRPOINT(NO.BDE.UNIT(BDE(T)))
          LET ADDR.STORE = ARRPOINT(NO.DIV.UNIT(DIV(T)))
          CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME) YIELDING MSG
          LET DEST.CODE(MSG) = DIV.LEVEL
          LET MSG.TYPE(MSG) = BDE.DIV.REQ.PERM.TO.MOVE
          SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) IN COMO3.DISTR(PARS4,PARS5,9) UNITS
          RETURN
     ELSE
          IF COM.PRINT GT 15
               PRINT 1 LINE WITH BDE(T) AND TIME.V THUS
          BDE(**) IS RESTRICTED AND ORDERED THE REQUESTING BN NOT TO MOVE AT *****.**
          ALWAYS
     LET FLAG = 20
     CALL EXIT(ORDER,T,CRIT,P.Z.TIME,FLAG)
     LET ADDR.STORE = ADRSEE
     LET T = ADRSEE
     CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME) YIELDING MSG
     LET DEST.CODE(MSG) = BN.LEVEL
     LET MSG.TYPE(MSG) = BDE.DIR.BN.NOT.TO.MOVE
     SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) IN COMO4.DISTR(PAR56,PAR57,7) UNITS
     RETURN
     END
```

APPENDIX L

```
ROUTINE DEC.DIV

IF COM.PRINT GT 20
     PRINT 1 LINE WITH TIME.V THUS
     ENTERED DEC.DIV ROUTINE AT TIME.V = *****.***
ALWAYS
LET TEXT(*) = TEXT.POINTER
LET PHAROW = TEXT(1)
LET ORDER = TEXT(2)
LET T = TEXT(3)
LET CRIT = TEXT(4)
LET P.Z.TIME = TEXT(5)
LET TEXT(*)=0
IF BDEREQST(BDE(T))=0 AND TYPE EQ BDE.DIV.REQ.PERM.TO.MOVE
     LET CRIT=5
     LET DIVCUR(DIV(T))=DIVCUR(DIV(T)) + BDEWT(BDE(T))
     LET DIVREQST(BDE(T))=1
ALWAYS
     IF COM.PRINT GT 15
          PRINT 1 LINE WITH DIV(T) AND TIME.V THUS
     DIV(**) NOT RESTRICTED AND GAVE BDES PERMISSION TO MOVE AT *****.***
     ALWAYS
     FOR EACH BDE.COMMANDER IN DIVISION(DIV(T)), DO
          LET FLAG=22
          LET T=RARPOINT(NO.BDE.UNIT(BDE.COMMANDER))
          LET ADDR.STORE = T
          CALL GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME) YIELDING MSG
          LET DEST.CODE(MSG) = BDE.LEVEL
          LET MSG.TYPE(MSG)=DIV.GAVE.BDE.PERM.TO.MOVE
          SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) IN COM20.DISTR(PAR71,PAR72,7) UNITS
     LOOP
RETURN
END
```

APPENDIX M

```
ROUTINE EXIT(ORDER,T,CRIT,P.Z.TIME,FLAG)

IF CRIT NE 0 AND CRIT LE 3
     LET CRIT = CRIT - 1
     CALL AIR.COMMO(T,CRIT)
     LET CRIT = CRIT + 1
ALWAYS
IF (ORDER=0  OR  ORDER=2) AND P.Z.TIME=NO
     SCHEDULE A PHAZ.CHK IN 60 UNITS LET P.Z.TIME=YES
ALWAYS
IF ORDER=2 LET ORDER=NO
ALWAYS
IF COM.PRINT GT 20 PRINT 1 LINE WITH TIME.V AND FLAG THUS
     OUT OF DECISION AT *****.*** THROUGH EXIT **
ALWAYS
RETURN
END
```

APPENDIX N

```
ROUTINE GEN.MOVE.DECISION.MSG(PHAROW,ORDER,T,CRIT,P.Z.TIME) YIELDING MSG

CREATE A MESSAGE CALLED MSG
RESERVE TEXT(*) AS 5
LET MSG.TEXT(MSG)=TEXT(*)
LET TEXT(1)=PHAROW
LET TEXT(2)=ORDER
LET TEXT(3)=T
LET TEXT(4)=CRIT
LET TEXT(5)=P.Z.TIME
LET ADDRESSEE(MSG)=ADDR.STORE
LET LENGTH.MSG(MSG)=30.0
LET LST.SEND.TIME(MSG)=TIME.V + 1200.0
LET MSG.NO(MSG) = NXT.MSG.NO(FMTANK.LIST(ORIG))
ADD 1 TO NXT.MSG.NO(FMTANK.LIST(ORIG))
IF COM.PRINT GT 15
     SKIP 1 LINE
     PRINT 1 LINE WITH TIME.V AS FOLLOWS
*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V=******.***
     PRINT 1 DOUBLE LINE WITH MSG.NO(MSG), ORIG, ADDRESSEE(MSG), TEXT(*),
          PHAROW, ORDER, T, CRIT, AND FLAG THUS
     MSG.NO ******   FROM ********  TO ********  TEXT ********  PHAROW **  ORDER
 **  T ********  CRIT ***              RETURN **
ALWAYS
LET TEXT(*) =0
LET FLAG = 0
RETURN
END
```

APPENDIX O

```
ROUTINE COM.MAIN
RESERVE DUMMY(*) AS 1
CALL BUILDNETS
CALL CHECKNETS
CALL BNETTBL
CALL COMMGEAR
CALL COMMCHECK
CALL INITIALIZE
LET FMNET = F.ETHER
SCHEDULE A DUMP.MAILBAG IN 600.0 UNITS
SCHEDULE A MSG.GEN(FMNET) IN COM01.DISTR(PAR50,PAR51,1) UNITS
IF COM.PRINT GT 25
     START NEW PAGE
     LIST ATTRIBUTES OF EACH B.MVR.COR
     IF LINE.V GT 20
          START NEW PAGE
     ALWAYS
     LIST ATTRIBUTES OF EACH UNIT IN BLUE.ALIVE
ALWAYS
START NEW PAGE
RETURN
END
```

APPENDIX P

```
ROUTINE BUILDNETS
DEFINE I, J, K, NALTFR, NETNO, NFMN, NONETS, NORELAY, NPRI, NRSUB, NTCK, TKALT,
     AND TKHOLD AS INTEGER VARIABLES

  READ  NONETS, NORELAY
  IF NONETS IS EQUAL TO ZERO,
     GO TO X3
  OTHERWISE
    FOR I = 1 TO NONETS, DO
      LET NETNO = 0  ''DONE TO AID ERROR IDENTIFICATION''
      READ NFMN, NRSUB, NALTFR
      READ NTCK AND NPRI
      IF NTCK IS NOT EQUAL TO NFMN,
        GO TO ERROR
      OTHERWISE
      CREATE AN FMNET
      FILE FMNET IN THE ETHER
        FOR J=1 TO NRSUB, DO
          READ  TKHOLD
          CREATE A SUBSCRIBER
          LET TFC.WAITING(SUBSCRIBER) = 0
          LET OWN.NET(SUBSCRIBER) = FMNET
          LET OWN.TANK(SUBSCRIBER) = ARRPOINT(TKHOLD)
          FILE SUBSCRIBER IN CEOI.LIST(FMNET)
        LOOP
      LET NUM.NET(FMNET) = NFMN
      LET FM.IDLE(FMNET) = IDLE
      LET FMPRI.FREQ(FMNET) = NPRI
      IF NALTFR IS EQUAL TO ZERO,
        LET FMALT.FREQ(FMNET) = 0
        GO TO X4
      OTHERWISE
        READ NETNO
        IF NETNO IS NOT EQUAL TO NFMN,
          GO TO ERROR
        OTHERWISE
        RESERVE FM.PUSHES(*) AS (NALTFR + 1)
        LET FMALT.FREQ(FMNET) = FM.PUSHES(*)
        LET FM.PUSHES(1) = NALTFR
        FOR K=1 TO NALTFR, DO
          READ TKALT
          LET FM.PUSHES(K+1) = TKALT
        LOOP
'X4'  ''CONTINUE''
      LET FM.PUSHES(*) = 0
    LOOP
'X3'  ''CONTINUE''
    RETURN

'ERROR'  ''CONTINUE''
```

```
              PRINT 2 LINES WITH NFMN, NTCK AND NETNO LIKE THIS
         ERROR, NET NUMBER ****, SUBS. NUMBER ****,
         AND ALT.FREQ NET NUMBER ****  ARE INCONSISTENT
            SKIP 1 LINE
            PRINT 1 LINE WITH NONETS AND NORELAY LIKE THIS
         NONETS=****, NOREALY=****
            PRINT 1 LINE WITH NFMN, NRSUB, NALTFR LIKE THIS
         NFMN=****, NRSUB=****, NALTFR=****
            PRINT 1 LINE WITH NTCK,NPRI,TKHOLD LIKE THIS
         NTCK=****, NPRI=****,TKHOLD= ****
              STOP
     END
```

APPENDIX Q

```
ROUTINE CHECKNETS
DEFINE I AND NFMALT AS INTEGER VARIABLES

  FOR EACH FMNET IN THE ETHER, DO
    IF LINE.V GT 57 START NEW PAGE
    ALWAYS
    PRINT 1 LINE WITH NUM.NET(FMNET) LIKE THIS
     FOR NET NUMBER ****
    SKIP 1 LINE
    PRINT 1 LINE WITH FMPRI.FREQ(FMNET) LIKE THIS
     PRIMARY FREQUENCY IS  ********* KHZ

    IF FMALT.FREQ(FMNET) IS NOT EQUAL TO ZERO,
      LET FM.PUSHES(*) = FMALT.FREQ(FMNET)
      LET NFMALT = FM.PUSHES(1)
      PRINT 1 LINE WITH NFMALT LIKE THIS
     THERE ARE ******* ALTERNATE FREQUENCIES
      FOR I = 1 TO NFMALT, DO
        PRINT 1 LINE WITH FM.PUSHES(I+1) LIKE THIS
     **********
      LOOP
    OTHERWISE
      PRINT 1 LINE LIKE THIS
     THERE ARE NO ALTERNATE FREQUENCIES
    ALWAYS
    PRINT 1 LINE WITH N.CEOI.LIST(FMNET) LIKE THIS
     THERE ARE ***** SUBSCRIBERS IN THE NET.
        SKIP 3 LINES
    LOOP
    RETURN
    END
```

APPENDIX R

```
ROUTINE BNETTBL
DEFINE I AND J AS INTEGER VARIABLES

  RESERVE NETTBL(*,*) AS N.ETHER BY *
  RESERVE RPOINT(*,*) AS N.ETHER BY *
  LET I = 1
  FOR EVERY FMNET IN THE ETHER, DO
    RESERVE NETTBL(I,*) AS N.CEOI.LIST(FMNET)
    RESERVE RPOINT(I,*) AS N.CEOI.LIST(FMNET)
    LET J = 1
    FOR EVERY SUBSCRIBER IN CEOI.LIST(FMNET), DO
      LET NETTBL(I,J) = TK.ID(OWN.TANK(SUBSCRIBER))
      LET RPOINT(I,J) = SUBSCRIBER
      LET J = J + 1
    LOOP
    LET I = I + 1
  LOOP

  IF LINE.V GT 62 START NEW PAGE
  ALWAYS
  PRINT 1 LINE LIKE THIS
     THE ITH ROW OF ARRAY NETTBL CONTAINS THE SUBSCRIBERS IN THE ITH RADIO NET
  LIST NETTBL
  SKIP 2 LINES
  IF LINE.V GT 68 START NEW PAGE
  ALWAYS
  PRINT 1 DOUBLE LINE LIKE THIS
     THE ITH ROW OF THE ARRAY RPOINT CONTAINS THE ADDRESS POINTERS OF THE SUBSCR
IBERS IN THE ITH RADIO NET
  LIST RPOINT
  SKIP 2 LINES
  RELEASE RPOINT(*,*)
  RETURN
END
```

APPENDIX S

```
ROUTINE COMMGEAR
DEFINE I, J, NORAD, NRADIO.TANKS, AND TKNO AS INTEGER VARIABLES

  READ NRADIO.TANKS

  FOR I = 1 TO NRADIO.TANKS, DO
    READ TKNO, NORAD
    CREATE A MYSET
    LET  TANK = RARPOINT(TKNO)
    LET  FMTANK.LIST(TANK) = MYSET
    FOR J = 1 TO NORAD, DO
      CREATE AN RT.CONFIG
      FILE RT.CONFIG IN RAD.LIST(MYSET)
      READ RAD.TYPE(RT.CONFIG) AND MODE.OPERATION(RT.CONFIG)
      READ ANT.TYPE(RT.CONFIG)
      IF ANT.TYPE(RT.CONFIG) IS LESS THAN ZERO,
          ''DIRECTIONAL ANTENNA INPUT''
        PRINT 3 LINES WITH TKNO, NORAD, J, RAD.TYPE(RT.CONFIG),
          MODE.OPERATION(RT.CONFIG) AND ANT.TYPE(RT.CONFIG) LIKE THIS
     ON TANK ****, WITH A TOTAL OF **** RADIOS, FOR RADIO SET
     NUMBER ****, OF TYPE ****, MODE ****, AND ANTENNA ****
     -----THIS SET OPERATES A DIRECTIONAL ANTENNA
        STOP
      ALWAYS
      READ REM.ANT(RT.CONFIG)
      IF  REM.ANT(RT.CONFIG) IS NOT EQUAL TO ZERO,
          ''REMOTED ANTENNA INPUT''
        PRINT 3 LINES WITH TKNO, NORAD, J, RAD.TYPE(RT.CONFIG),
          MODE.OPERATION(RT.CONFIG) AND ANT.TYPE(RT.CONFIG) LIKE THIS
     ON TANK ****, WITH A TOTAL OF **** RADIOS, FOR RADIO SET
     NUMBER ****, OF TYPE **** AND MODE ****, WITH ANTENNA ****
     -----THIS SET OPERATES WITH A REMOTED ANTENNA
        STOP
      REGARDLESS
    LOOP
  LOOP

      ''USE NET STRUCTURE INPUT PREVIOUSLY TO ASSIGN EQUIPMENTS''
          ''TO NETS''

    IF LINE.V GT 30 START NEW PAGE
    ALWAYS
    PRINT 4 LINES AS FOLLOWS
     RARPOINT(TK.ID(TANK)) = TANK
     FMTANK.LIST(TANK) = MYSET(TANK)


  FOR EVERY FMNET IN THE ETHER, DO
    PRINT 1 LINE WITH NUM.NET(FMNET) AND FMNET AS FOLLOWS
     NUM.NET(FMNET) = ***     FMNET POINTER = ********
```

```
      SKIP 1 LINE
      FOR EVERY SUBSCRIBER IN CEOI.LIST(FMNET), DO
        LET TANK = OWN.TANK(SUBSCRIBER)
        IF FMTANK.LIST(TANK) EQUALS ZERO,
          GO TO ERROR1
        OTHERWISE
        LET MYSET = FMTANK.LIST(TANK)
        LET NXT.MSG.NO(MYSET) = (TK.ID(OWN.TANK(SUBSCRIBER)) * 1000) + 1
        FOR EVERY RT.CONFIG IN RAD.LIST(MYSET), DO
          IF OWN.SUBSCRIBER(RT.CONFIG) EQUALS ZERO,
            GO TO JOIN.NET
          OTHERWISE
        LOOP
        GO TO ERROR2

  'JOIN.NET'  ''PLACE THIS EQUIPMENT ON THE NET''
        LET OWN.SUBSCRIBER(RT.CONFIG) = SUBSCRIBER
        PRINT 1 DOUBLE LINE WITH TK.ID(TANK), TANK, FMTANK.LIST(TANK), RT.CONFIG,
                AND OWN.SUBSCRIBER(RT.CONFIG) AS FOLLOWS
       TK.ID(TANK)=*****         TANK=********          FMTANK.LIST(TANK)=********
       RT.CONFIG=********         SUBSCRIBER=********

      LOOP
      SKIP 1 LINE
    LOOP
      SKIP 5 LINES

  RETURN

  'ERROR1'  ''ERROR WHEN TANK HAS NO RADIOS''
    PRINT 1 LINE WITH TK.ID(TANK) AND NUM.NET(FMNET) LIKE THIS
       ERROR, TANK **** ON NET **** HAS NO RADIOS
    STOP

  'ERROR2'  ''ERROR WHEN TANK HAS TOO FEW RADIOS FOR ASGD. NETS''
    PRINT 1 LINE WITH TK.ID(TANK) AND NUM.NET(FMNET) LIKE THIS
       ERROR, TANK **** HAS TOO FEW RADIOS TO BE ON NET ****
    STOP

  END
```

APPENDIX T

```
ROUTINE COMMCHECK

FOR EACH TANK IN BLUE.ALIVE, DO
  IF LINE.V GT 60 START NEW PAGE
  ALWAYS
  IF FMTANK.LIST(TANK) IS EQUAL TO ZERO,
    PRINT 2 LINES WITH TK.ID(TANK), X.CURRENT(TANK),
      Y.CURRENT(TANK) AND Z.CURRENT(TANK) LIKE THIS
     TANK ****
     LOCATED AT ****.*, ****.*, ****.* HAS NO COMMO EQUIPMENT
    GO TO X1

  OTHERWISE
  PRINT 2 LINES WITH TK.ID(TANK), X.CURRENT(TANK),
    Y.CURRENT(TANK) AND Z.CURRENT(TANK) LIKE THIS
     TANK ****
     LOCATED AT ****.*, ****.*, ****.* HAS THE FOLLOWING COMMO
  SKIP 1 LINE

  LET MYSET = FMTANK.LIST(TANK)
  FOR EACH RT.CONFIG IN RAD.LIST(MYSET), DO
    PRINT 2 LINES WITH RAD.TYPE(RT.CONFIG) AND
      MODE.OPERATION(RT.CONFIG) LIKE THIS
     RADIO TYPE = ***
     -----MODE OF OPERATION = ****
    IF ANT.TYPE(RT.CONFIG) IS LESS THAN ZERO
      PRINT 1 LINE LIKE THIS
     -----THIS SET OPERATES A DIRECTIONAL ANTENNA
      STOP
    OTHERWISE
      PRINT 1 LINE WITH ANT.TYPE(RT.CONFIG) LIKE THIS
     ANTENNA TYPE = ****
    IF REM.ANT(RT.CONFIG) IS NOT EQUAL TO ZERO
      PRINT 1 LINE LIKE THIS
     -----THIS SET OPERATES A REMOTED ANTENNA
        ''GO TO NO.NET''
      STOP
    OTHERWISE
      PRINT 1 LINE WITH X.CURRENT(TANK), Y.CURRENT(TANK),
        AND Z.CURRENT(TANK) LIKE THIS
     ANTENNA LOCATION: X = *****.*, Y = *****.*, Z = *****.*

'NO.NET'   ''CONTINUE''
    PRINT 1 LINE WITH NUM.NET(OWN.NET(OWN.SUBSCRIBER(RT.CONFIG)))
      LIKE THIS
     -----THIS SET IS TUNED TO NET NUMBER ****
    SKIP 1 LINE
  LOOP

'X1'  ''CONTINUE''
```

```
  SKIP 5 LINES

LOOP
RETURN
END
```

APPENDIX U

```
ROUTINE INITIALIZE
DEFINE I, J, K, AND NPRI AS INTEGER VARIABLES

READ PAR00, PAR01, PAR02, PAR03, PAR04, PAR05, PAR06, PAR07, PAR08, AND PAR09
READ PAR10 AND PAR11
READ PAR50, PAR51, PAR52, PAR53, PAR54, PAR55, PAR56, PAR57, PAR58, AND PAR59
READ PAR60, PAR61, PAR62, PAR63, PAR64, PAR65, PAR66, PAR67, PAR68, AND PAR69
READ PAR70, PAR71, AND PAR72
START NEW PAGE
IF COM.PRINT GT 10
     CALL STAT.DUMP
     CALL SSTAT.DUMP
ALWAYS
READ CO.LO.BD AND CO.UP.BD
READ BN.LO.BD AND BN.UP.BD
READ BDE.LO.BD AND BDE.UP.BD
READ DIV.LO.BD AND DIV.UP.BD
RETURN
END
```

APPENDIX V

```
ROUTINE STAT.DUMP

LIST ATTRIBUTES OF EACH COMPANY.COMMANDER
LIST ATTRIBUTES OF EACH BN.COMMANDER
LIST ATTRIBUTES OF EACH BDE.COMMANDER
LIST ATTRIBUTES OF EACH DIV.COMMANDER
SKIP 2 LINES
RETURN
END
```

APPENDIX W

```
ROUTINE SSTAT.DUMP
DEFINE I AND J AS INTEGER VARIABLES

     FOR I = 1 TO BBDE, DO
          IF LINE.V GT 70 START NEW PAGE
          ALWAYS
          PRINT 1 LINE WITH I AS FOLLOWS
     BRIGADE **
          PRINT 3 LINES AS FOLLOWS
                    BDEMSN      BDECUR    BDEWT     BDELO     BDEGO
--------------------------------------------------------------------------------
     PHZLINES    .
               FOR J=1 TO PHZLINES, DO
                    PRINT 1 LINE WITH J, BDECORD(I,J,1), BDECORD(I,J,2),
                        BDECORD(I,J,3), BDECORD(I,J,4), AND BDECORD(I,J,5) THUS
              ** .  *****     *****     *****     *****     *****
               LOOP
     SKIP 2 LINES
     LOOP
     FOR I=1 TO BBN, DO
          IF LINE.V GT 70 START NEW PAGE
          ALWAYS
          PRINT 1 LINE WITH I AS FOLLOWS
     BATTALION **
          PRINT 3 LINES AS FOLLOWS
                      BNMSN     BNCUR     BNWT      BNLO      BNGO
--------------------------------------------------------------------------------
     PHZLINES    .
               FOR J=1 TO PHZLINES, DO
                    PRINT 1 LINE WITH J, BNCORD(I,J,1), BNCORD(I,J,2),
                        BNCORD(I,J,3), BNCORD(I,J,4), AND BNCORD(I,J,5) THUS
              ** .  *****     *****     *****     *****     *****
               LOOP
     SKIP 2 LINES
     LOOP
     FOR I=1 TO BLCOMP, DO
          IF LINE.V GT 70 START NEW PAGE
          ALWAYS
          PRINT 1 LINE WITH I AS FOLLOWS
     COMPANY **
          PRINT 3 LINES AS FOLLOWS
                      CMSN      COWT
--------------------------------------------------------------------------------
     PHZLINES    .
               FOR J=1 TO PHZLINES, DO
                    PRINT 1 LINE WITH J, COCORD(I,J,1), AND COCORD(I,J,2) THUS
              ** .  *****     *****
               LOOP
```

```
        SKIP 2 LINES
        LOOP
    RETURN
    END
```

APPENDIX X

```
UPON COMMO.ATTEMPT(ORIG,MSG)

  IF COM.PRINT GT 20
     PRINT 1 LINE WITH TIME.V THUS
     ENTERED COMMO.ATTEMPT EVENT AT TIME.V=******.***
  ALWAYS
  LET CALLNET = 0
  LET CALLSUB = 0
  LET CALLRT = 0
  LET ADRSEE = ADDRESSEE(MSG)
  LET OR.TANK = TK.ID(ORIG)
  LET DE.TANK = TK.ID(ADRSEE)

''CHECK THAT BOTH TANKS HAVE RADIOS''
  IF FMTANK.LIST(ORIG) IS EQUAL TO ZERO,
    GO TO ERR1
  OTHERWISE
  IF FMTANK.LIST(ADRSEE) IS EQUAL TO ZERO,
    GO TO ERR1
  OTHERWISE

''CHECK THAT DESTINATION AND ORIGINATOR ARE ON SAME NET''
  LET MYSET = FMTANK.LIST(ORIG)
  FOR EVERY RT.CONFIG IN RAD.LIST(MYSET), DO
    LET FMNET = OWN.NET(OWN.SUBSCRIBER(RT.CONFIG))
    FOR EVERY SUBSCRIBER IN CEOI.LIST(FMNET), DO
      IF OWN.TANK(SUBSCRIBER) = ADRSEE,
        LET CALLRT = RT.CONFIG
        LET CALLSUB = SUBSCRIBER
        LET CALLNET = FMNET
        IF FM.IDLE(FMNET) IS EQUAL TO IDLE,
          GO TO TRY.CALL
        OTHERWISE
      REGARDLESS
    LOOP
  LOOP
  IF CALLRT IS NOT EQUAL TO ZERO,
    GO TO NET.BUSY
  OTHERWISE
    GO TO ERR2

''TRY TO CALL THE DISTANT STATION''
'TRY.CALL'
  IF COM.PRINT GT 25
     PRINT 1 LINE WITH MSG, CALLNET, ORIG, AND ADRSEE AS FOLLOWS
     -----MSG=********  FMNET=********  ORIGINATOR=********  ADDRESSEE=********
  ALWAYS
  LET FM.IDLE(CALLNET) = BUSY
  LET IN.USE(CALLRT) = BUSY
  LET NET.NO = NUM.NET(CALLNET)
```

```
      LET MYSET = FMTANK.LIST(ADRSEE)
      FOR EVERY RT.CONFIG IN RAD.LIST(MYSET), DO
        IF OWN.NET(OWN.SUBSCRIBER(RT.CONFIG)) = CALLNET,
          GO TO KEY.MIKE
        OTHERWISE
      LOOP
      GO TO NO.ANS


    ''CHECK IF COMMO IS TECHNICALLY FEASIBLE''
    'KEY.MIKE'
      LET RECRT = RT.CONFIG
      IF COM.PRINT GT 25
        PRINT 1 LINE WITH CALLRT AND RECRT AS FOLLOWS
         -----TRANSMITTER/CALLRT=********          RECEIVER/RECRT=********
      ALWAYS
      CALL TECH.COMMO(CALLRT,RECRT) YIELDING ABILITY

      IF ABILITY = ROGER,
        LET IN.USE(RECRT) = BUSY
        LET DELT = LENGTH.MSG(MSG)
       CALL EW.ROUTINE(CALLRT,RECRT) YIELDING ABILITY
      IF ABILITY = JAMMED
         IF COM.PRINT GT 20
              PRINT 1 DOUBLE LINE WITH MSG.NO(MSG), MSG.TYPE(MSG), NUM.NET(FMNET),
                   TK.ID(ORIG), AND TK.ID(ADDRESSEE(MSG)) AS FOLLOWS
         MSG.NO ****** OF TYPE *** ON NET *** FROM TANK *** TO TANK ***
            +++   JAMMED   +++
              SKIP 1 LINE
              ALWAYS
        SCHEDULE A CHANGE.FREQ(CALLRT,CALLNET,MSG) IN COM15.DISTR(PAR69,PAR70,2)
              UNITS
        RETURN
      OTHERWISE
         IF COM.PRINT GT 20
              PRINT 1 DOUBLE LINE WITH MSG.NO(MSG), MSG.TYPE(MSG), NUM.NET(FMNET),
                   TK.ID(ORIG), AND TK.ID(ADDRESSEE(MSG)) AS FOLLOWS
         MSG.NO ****** OF TYPE *** ON NET *** FROM TANK *** TO TANK ***
            +++   NORMAL TRANSMISSION   +++
         ALWAYS
        SCHEDULE AN END.XSMN(CALLRT,RECRT,CALLNET,MSG) IN DELT UNITS
        RETURN
      OTHERWISE

      IF ABILITY = NEGATIVE
    ''DISTANT STATION DOES NOT RESPOND TO CALL''
    'NO.ANS'
         IF COM.PRINT GT 20
              PRINT 1 DOUBLE LINE WITH MSG.NO(MSG), MSG.TYPE(MSG), NUM.NET(FMNET),
                   TK.ID(ORIG), AND TK.ID(ADDRESSEE(MSG)) AS FOLLOWS
```

```
           MSG.NO ****** OF TYPE *** ON NET *** FROM TANK *** TO TANK ***
              +++  NO RESPONSE TO CALL  +++
                SKIP 1 LINE
                ALWAYS
      LET TFC.WAITING(CALLSUB) = WAITING
      SCHEDULE A NO.CONTACT(CALLAT,CALLNET,MSG) IN COM14.DISTR(PAR67,PAR68,2) UNITS
      RETURN
      OTHERWISE

      IF ABILITY = MARGINAL
        LET IN.USE(RECAT) = BUSY
        LET DELT = 2*LENGTH.MSG(MSG)
       CALL EW.ROUTINE(CALLAT,RECAT) YIELDING ABILITY
      IF ABILITY = JAMMED
         IF COM.PRINT GT 20
              PRINT 1 DOUBLE LINE WITH MSG.NO(MSG), MSG.TYPE(MSG), NUM.NET(FMNET),
                   TK.ID(ORIG), AND TK.ID(ADDRESSEE(MSG)) AS FOLLOWS
         MSG.NO ****** OF TYPE *** ON NET *** FROM TANK *** TO TANK ***
            +++  JAMMED  +++
              SKIP 1 LINE
              ALWAYS
        SCHEDULE A CHANGE.FREQ(CALLAT,CALLNET,MSG) IN COM15.DISTR(PAR69,PAR70,2)
              UNITS
        RETURN
      OTHERWISE
         IF COM.PRINT GT 20
              PRINT 1 DOUBLE LINE WITH MSG.NO(MSG), MSG.TYPE(MSG), NUM.NET(FMNET),
                   TK.ID(ORIG), AND TK.ID(ADDRESSEE(MSG)) AS FOLLOWS
         MSG.NO ****** OF TYPE *** ON NET *** FROM TANK *** TO TANK ***
            +++  MARGINAL TRANSMISSION  +++
              ALWAYS
        SCHEDULE AN END.XSMN(CALLAT,RECAT,CALLNET,MSG) IN DELT UNITS
        RETURN
      OTHERWISE
        GO TO ERR3

    'NET.BUSY'
    ''FILE MESSAGE AND CONTINUE WHEN NET IS BUSY''
         IF COM.PRINT GT 20
      PRINT 1 DOUBLE LINE WITH MSG.NO(MSG), MSG.TYPE(MSG), NUM.NET(FMNET),
              TK.ID(ORIG), AND TK.ID(ADDRESSEE(MSG)) AS FOLLOWS
         MSG.NO ****** OF TYPE *** ON NET *** FROM TANK *** TO TANK ***
            +++  NET BUSY, MSG FILED IN MAILBAG  +++
      SKIP 1 LINE
      ALWAYS
      FILE THIS MSG IN MAILBAG(MYSET)
      LET TFC.WAITING(CALLSUB) = WAITING
      RETURN

    ''ERROR ROUTINES ''
```

```
'ERR1'
  PRINT 2 LINES WITH TIME.V, OR.TANK AND DE.TANK LIKE THIS
     AT *****.**, TANK **** TRIED TO SEND A MESSAGE TO TANK ****
     ONE OF THESE DOES NOT HAVE A RADIO
  STOP

'ERR2'
  PRINT 2 LINES WITH TIME.V, OR.TANK, AND DE.TANK LIKE THIS
     AT *****.**, TANK **** TRIED TO SEND A MESSAGE TO TANK ****
     THESE TANKS ARE NOT ON A COMMON NET
    STOP

'ERR3'
  PRINT 2 LINES WITH TIME.V, OR.TANK, AND DE.TANK LIKE THIS
     AT *****.**, TANK **** TRIED TO CONTACT TANK **** ON THE FM
     AN ERRONEOUS ABILITY VALUE WAS RETURNED BY TECH.COMMO
  STOP
END
```

APPENDIX Y

```
ROUTINE TECH.COMMO(CALLRT,RECRT) YIELDING ABILITY

IF COM.PRINT GT 25
     PRINT 1 DOUBLE LINE WITH TK.ID(OWN.TANK(OWN.SUBSCRIBER(CALLRT))), CALLRT,
          TK.ID(OWN.TANK(OWN.SUBSCRIBER(RECRT))), AND RECRT AS FOLLOWS
     -----MSG FROM TANK *** TRANSMITTER=********  TO TANK *** RECEIVER=********
        +++   TECH.COMMO   +++
ALWAYS
LET ABILITY = COM13.DISTR(PAR04,PAR03,2)
RETURN
END


ROUTINE EW.ROUTINE(CALLRT,RECRT) YIELDING ABILITY

IF COM.PRINT GT 20
     PRINT 1 LINE WITH TIME.V AS FOLLOWS
     +++++  ENTERED EW ROUTINE AT TIME = *****.*****  +++++
     SKIP 1 LINE
ALWAYS
LET ABILITY = COM13.DISTR(PAR04,PAR05,2)
RETURN
END
```

APPENDIX Z

```
UPON CHANGE.FREQ(CALLRT,CALLNET,MSG)

  LET FM.IDLE(CALLNET) = IDLE
  LET IN.USE(CALLRT) = IDLE
  LET ORIG = OWN.TANK(OWN.SUBSCRIBER(CALLRT))
  IF COM.PRINT GT 20
       PRINT 1 DOUBLE LINE WITH TIME.V, TK.ID(ORIG) AND
              NUM.NET(CALLNET) LIKE THIS
     TIME.V  *****.**    TANK ***,                          ON NET ***
        +++   FREQUENCY CHANGE   +++
     SKIP 1 LINE
  ALWAYS

  SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) NOW

RETURN
END
```

APPENDIX AA

```
ROUTINE EW.ROUTINE(CALLAT,RECAT) YIELDING ABILITY
DEFINE ABILITY, CALLAT, AND RECAT AS INTEGER VARIABLES

IF COM.PRINT GT 20
     PRINT 1 LINE WITH TIME.V AS FOLLOWS
     +++++  ENTERED EW ROUTINE AT TIME = *****.*****  +++++
     SKIP 1 LINE
ALWAYS
LET ABILITY = COM13.DISTR(PAR04,PAR05,2)
RETURN
END
```

APPENDIX BB

```
UPON NO.CONTACT(CALLRT.CALLNET,MSG)

IF COM.PRINT GT 25
     PRINT 1 LINE WITH TIME.V THUS
     NO CONTACT EVENT ENTERED AT TIME.V=*****.***
     SKIP 1 LINE
ALWAYS
  LET FM.IDLE(CALLNET) = IDLE
  LET IN.USE(CALLRT) = IDLE

  CALL SIEZE.NET(CALLNET)
  LET MYSET = FMTANK.LIST(OWN.TANK(OWN.SUBSCRIBER(CALLRT)))
  FILE THIS MSG IN MAILBAG(MYSET)

RETURN
END
```

APPENDIX CC

```
ROUTINE SIEZE.NET(CALLNET)
DEFINE I, IK, AND SUBREC AS INTEGER VARIABLES

IF COM.PRINT GT 20
     PRINT 1 LINE WITH TIME.V AS FOLLOWS
     ENTERED SIEZE.NET ROUTINE AT TIME.V=*****.***
ALWAYS
RESERVE TARRAY(*) AS 2*N.CEOI.LIST(CALLNET)
LET I=1
FOR EVERY SUBSCRIBER IN CEOI.LIST(CALLNET), DO
     LET MYSET = FMTANK.LIST(OWN.TANK(SUBSCRIBER))
     IF N.MAILBAG(MYSET) = 0, GO TO LOOP
     OTHERWISE
     FOR EVERY MESSAGE IN MAILBAG(MYSET), DO
          LET ADRSEE = ADDRESSEE(MESSAGE)
          FOR EVERY SUBREC IN CEOI.LIST(CALLNET), DO
               IF OWN.TANK(SUBREC) = ADRSEE, GO TO MATCH
               OTHERWISE
          LOOP
     LOOP
GO TO LOOP

'MATCH'

LET TARRAY(I) = OWN.TANK(SUBSCRIBER)
LET TARRAY(I+1) = MESSAGE
LET I = I+2

'LOOP'

LOOP

IF I = 1
     IF COM.PRINT GT 20
          PRINT 1 DOUBLE LINE WITH TIME.V AND NUM.NET(CALLNET) AS FOLLOWS
     TIME.V  *****.***                                           NET ***
        +++   EMPTY MAILBAG   +++
          SKIP 1 LINE
     ALWAYS
     RELEASE TARRAY(*)
     RETURN
OTHERWISE
LET RNI = REAL.F((I-1)/2) - 0.0001
LET IK = TRUNC.F(COM12.DISTR(PAR66,RNI,8) + 1)
LET ORIG = TARRAY((2*IK) - 1)
LET MSG = TARRAY(2*IK)
LET MYSET = FMTANK.LIST(ORIG)
IF COM.PRINT GT 20
     PRINT 1 DOUBLE LINE WITH MSG.NO(MSG), MSG.TYPE(MSG), NUM.NET(CALLNET),
          TK.ID(ORIG), AND TK.ID(ADDRESSEE(MSG)) AS FOLLOWS
```

```
          MSG.NO ****** OF TYPE *** ON NET *** FROM TANK *** TO TANK ***
             +++ SIEZE NET   +++
          SKIP 1 LINE
      ALWAYS
      REMOVE THIS MSG FROM MAILBAG(MYSET)
      SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) NOW
      RELEASE TARRAY(*)
      RETURN
      END
```

```
UPON END.XSMN(CALLAT,RECAT,CALLNET,MSG)

  LET ORIG = OWN.TANK(OWN.SUBSCRIBER(CALLAT))
  LET ADRSEE = ADDRESSEE(MSG)
  LET TYPE = MSG.TYPE(MSG)
  LET TEXT.POINTER = MSG.TEXT(MSG)
  LET MSGNO = MSG.NO(MSG)
  LET DEST = DEST.CODE(MSG)
  IF COM.PRINT GT 15
       PRINT 1 LINE WITH TIME.V AS FOLLOWS
     END.XSMN EVENT ENTERED AT TIME.V=*****.***
       PRINT 1 DOUBLE LINE WITH MSG.NO(MSG), MSG.TYPE(MSG), NUM.NET(CALLNET),
          TK.ID(ORIG), AND TK.ID(ADDRESSEE(MSG)) AS FOLLOWS
     MSG.NO ****** OF TYPE *** ON NET *** FROM TANK *** TO TANK ***
        +++   END.XSMN COMPLETED   +++
  ALWAYS
  IF COM.PRINT GT 25
       PRINT 1 LINE WITH DEST.CODE(MSG), LENGTH.MSG(MSG), AND LST.SEND.TIME(MSG)
          THUS
     -----DEST.CODE ***  MSG.LENGTH ***.**  KILL.TIME ****.**
       PRINT 1 LINE WITH MSG, CALLNET, ORIG, AND ADRSEE AS FOLLOWS
     -----MSG=********  FMNET=********  ORIGINATOR=********  ADDRESSEE=********
       PRINT 1 LINE WITH CALLAT AND RECAT AS FOLLOWS
     -----TRANSMITTER/CALLAT=********          RECEIVER/RECAT=********
       PRINT 1 LINE WITH TEXT.POINTER AS FOLLOWS
     -----TEXT POINTER=********
  ALWAYS
  SKIP 1 LINE
  DESTROY THE MESSAGE CALLED MSG

  IF DEST LT 10 GO TO FINISHED
  ELSE
  IF CO.IS.DEST.OF.MSG CALL CO.MSG GO TO COMPLETED
  ELSE
  IF BN.IS.DEST.OF.MSG CALL BN.MSG GO TO COMPLETED
  ELSE
  IF BDE.IS.DEST.OF.MSG CALL BDE.MSG GO TO COMPLETED
  ELSE
  IF DIV.IS.DEST.OF.MSG CALL DIV.MSG GO TO COMPLETED
  ELSE
  PRINT 1 LINE AS FOLLOWS
     AN ERROR HAS OCCURRED IN THE TYPE CODE.
  RETURN

'COMPLETED'
  LET TEXT(*) = TEXT.POINTER
  RELEASE TEXT(*)

'FINISHED'
  LET FM.IDLE(CALLNET) = IDLE
```

```
    LET IN.USE(CALLRT) = IDLE
    LET IN.USE(RECRT) = IDLE
    SKIP 1 LINE

    CALL SIEZE.NET(CALLNET)

  RETURN
  END
```

APPENDIX EE

```
ROUTINE CO.MSG

IF COM.PRINT GT 20
     PRINT 1 LINE WITH TIME.V AS FOLLOWS
     EXECUTION OF CO.MSG ROUTINE AT TIME.V=*****.***
ALWAYS
IF TYPE LT 20 CALL DEC.CO GO TO EXIT1
ELSE
'EXIT1'
RETURN
END


ROUTINE BN.MSG

IF COM.PRINT GT 20
     PRINT 1 LINE WITH TIME.V AS FOLLOWS
     EXECUTION OF BN.MSG ROUTINE AT TIME.V=*****.***
ALWAYS
IF TYPE LT 30 CALL DEC.BN GO TO EXIT1
ELSE
'EXIT1'
RETURN
END


ROUTINE BDE.MSG

IF COM.PRINT GT 20
     PRINT 1 LINE WITH TIME.V AS FOLLOWS
     EXECUTION OF BDE.MSG ROUTINE AT TIME.V=*****.***
ALWAYS
IF TYPE LT 40 CALL DEC.BDE GO TO EXIT1
ELSE
'EXIT1'
RETURN
END


ROUTINE DIV.MSG

IF COM.PRINT GT 20
     PRINT 1 LINE WITH TIME.V THUS
     ENTERED DIV.MSG ROUTINE AT TIME.V = *****.***
ALWAYS
CALL DEC.DIV
RETURN
END
```

APPENDIX FF

```
UPON DUMP.MAILBAG

IF COM.PRINT GT 25
     PRINT 1 LINE WITH TIME.V THUS
     DUMP MAILBAG EVENT ENTERED AT TIME.V = *****.***
     SKIP 1 LINE
ALWAYS
  FOR EVERY TANK IN BLUE.ALIVE, DO
    IF FMTANK.LIST(TANK) IS EQUAL TO ZERO,
      GO TO LOOP
    OTHERWISE
      LET MYSET = FMTANK.LIST(TANK)
      IF N.MAILBAG(MYSET) EQ 0 GO TO LOOP
      ELSE
      FOR EVERY MESSAGE IN MAILBAG(MYSET), DO
        IF LST.SEND.TIME(MESSAGE) IS LESS THAN TIME.V,
          REMOVE THE MESSAGE FROM MAILBAG(MYSET)
          CALL ABORT.MSG(TANK,MESSAGE)
        REGARDLESS
      LOOP

'LOOP'
  LOOP

SKIP 1 LINE
SCHEDULE A DUMP.MAILBAG IN 600.0 UNITS

RETURN
END
```

APPENDIX GG

```
ROUTINE ABORT.MSG(TNK,MSG)

IF COM.PRINT GT 20
     PRINT 1 DOUBLE LINE WITH TIME.V, TK.ID(TNK), MSG.NO(MSG),
          MSG.TYPE(MSG), AND TK.ID( ADDRESSEE(MSG)) LIKE THIS
     TIME.V  *****.**    TANK ***, MSG.NO ******, TYPE **, TO TANK ***
        +++   ABORTED DUE TO KILL TIME   +++
     SKIP 1 LINE
ALWAYS

DESTROY THE MESSAGE CALLED MSG
RETURN
END
```

APPENDIX HH

```
UPON MSG.GEN(FMNET)
DEFINE ADRSEEK, ORT.TEMP, I, NUMBNET, ORT.TEMP, ORTK, AND RNL AS INTEGER
     VARIABLES

  IF N.CEOI.LIST(FMNET) LT 2
     PRINT 1 LINE WITH FMNET AS FOLLOWS
     THERE IS ONLY ONE SUBSCRIBER ON FMNET *****.*****
     GO TO LOOP1
  ELSE
  LET RNL = N.CEOI.LIST(FMNET)
  LET ORTK = COM08.DISTR(PAR02,RNL,2)
'DESTGEN'
  LET ADRSEEK = COM08.DISTR(PAR02,RNL,2)
  IF ADRSEEK IS EQUAL TO ORTK, GO TO DESTGEN
  OTHERWISE

  CREATE A MESSAGE CALLED MSG
  LET I = NUM.NET(FMNET)
  LET ADDRESSEE(MSG) = RARPOINT(NETTBL(I,ADRSEEK))
  LET LST.SEND.TIME(MSG) = TIME.V + 250.0
  LET LENGTH.MSG(MSG) = COM09.DISTR(PAR62,PAR63,2)
  LET ORIG = RARPOINT(NETTBL(I,ORTK))
  LET MSG.TYPE(MSG) = 0
  LET MSG.TEXT(MSG)=0
  LET DEST.CODE(MSG)=0
  LET MYSET = FMTANK.LIST(ORIG)
  LET MSG.NO(MSG) = NXT.MSG.NO(MYSET)
  ADD 1 TO NXT.MSG.NO(MYSET)
  FOR EVERY RT.CONFIG IN RAD.LIST(MYSET), DO
     IF OWN.NET(OWN.SUBSCRIBER(RT.CONFIG)) EQ FMNET LET ORT.TEMP=RT.CONFIG
     ALWAYS
  LOOP
  FOR EVERY RT.CONFIG IN RAD.LIST(FMTANK.LIST(ADDRESSEE(MSG))), DO
     IF OWN.NET(OWN.SUBSCRIBER(RT.CONFIG)) EQ FMNET LET ORT.TEMP=RT.CONFIG
     ALWAYS
  LOOP

  IF COM.PRINT GT 15
     PRINT 1 LINE WITH TIME.V THUS
     ENTERED MESSAGE GENERATOR AT TIME.V = *****.***
     PRINT 1 DOUBLE LINE WITH MSG.NO(MSG), MSG.TYPE(MSG), NUM.NET(FMNET),
          TK.ID(ORIG), AND TK.ID(ADDRESSEE(MSG)) AS FOLLOWS
     MSG.NO ****** OF TYPE *** ON NET *** FROM TANK *** TO TANK ***
        +++   GENERATED   +++
  ALWAYS
  IF COM.PRINT GT 25
     PRINT 1 LINE WITH DEST.CODE(MSG), LENGTH.MSG(MSG), AND LST.SEND.TIME(MSG)
          THUS
     -----DEST.CODE ***  MSG.LENGTH ***.**  KILL.TIME ****.**
     PRINT 1 LINE WITH MSG, FMNET, ORIG, AND ADDRESSEE(MSG) AS FOLLOWS
```

```
          -----MSG=********  FMNET=********  ORIGINATOR=********  ADDRESSEE=********
          PRINT 1 LINE WITH ORT.TEMP AND ORT.TEMP AS FOLLOWS
          -----TRANSMITTER/CALLAT=********          RECEIVER/RECAT=********
       ALWAYS
       SKIP 1 LINE
       SCHEDULE A COMMO.ATTEMPT(ORIG,MSG) NOW
     'LOOP1'
          LET NUMBNET=(COM10.DISTR(PAR03,N.ETHER,8) - 1)
          LET FMNET=F.ETHER
          FOR I = 1 TO NUMBNET, DO
               LET FMNET = S.ETHER(FMNET)
          LOOP
       SCHEDULE A MSG.GEN(FMNET) IN COMO1.DISTR(PAR50,PAR51,1) UNITS
       RETURN
     END
```

```
     ENTERED MESSAGE GENERATOR AT TIME.V =     56.604
     MSG.NO    2001  OF TYPE    0  ON NET   1  FROM TANK   2  TO TANK  21                  +++  GENERATED   +++

     CALLING DECISION AT TIME.V=   78.113  FCN= 7  CCL= 1  J= 19  T=  992520
22222-----STATUS OF ACFR=  2  OF PFARCW=  1 CAUSED A RETURN

     END.XSMN EVENT ENTERED AT TIME.V=   75.604
     MSG.NO    2001  OF TYPE    0  ON NET   1  FROM TANK   2  TO TANK  21                  +++  END.XSMN COMPLETED   +++

     ENTERED MESSAGE GENERATOR AT TIME.V =     75.627
     MSG.NO   15001  OF TYPE    0  ON NET   3  FROM TANK  15  TO TANK  23                  +++  GENERATED   +++

     ENTERED MESSAGE GENERATOR AT TIME.V =    105.551
     MSG.NO   24001  OF TYPE    0  ON NET   5  FROM TANK  24  TO TANK  25                  +++  GENERATED   +++

     CALLING DECISION AT TIME.V=  155.342  FCN= 6  CCL= 1  J= 12  T=  992968
******EXECUTING CCN.MOVE.DECISION.MSC ROUTINE AT TIME.V=  155.342
     MSG.NO   12001   FROM   992968  TO   992264  TEXT  1092176  PFARCW  1  ORDER  1  T   992968  CRIT   0            RETURN  1
     *** DECISION BASED ON RANGE TO ENEMY ******

     ENTERED MESSAGE GENERATOR AT TIME.V =    164.468
     MSG.NO   13001  OF TYPE    0  ON NET   3  FROM TANK  13  TO TANK  23                  +++  GENERATED   +++

     END.XSMN EVENT ENTERED AT TIME.V=  167.627
     MSG.NO   15001  OF TYPE    0  ON NET   3  FROM TANK  15  TO TANK  23                  +++  END.XSMN COMPLETED   +++

     END.XSMN EVENT ENTERED AT TIME.V=  181.551
     MSG.NO   24001  OF TYPE    0  ON NET   5  FROM TANK  24  TO TANK  25                  +++  END.XSMN COMPLETED   +++

     ENTERED MESSAGE GENERATOR AT TIME.V =    190.562
     MSG.NO   26001  OF TYPE    0  ON NET   5  FROM TANK  26  TO TANK  22                  +++  GENERATED   +++

     END.XSMN EVENT ENTERED AT TIME.V=  215.627
     MSG.NO   13001  OF TYPE    0  ON NET   3  FROM TANK  13  TO TANK  23                  +++  END.XSMN COMPLETED   +++

     CALLING DECISION AT TIME.V=  241.706  FCN= 3  CCL= 1  J= 20  T=  992456
******EXECUTING CCN.MOVE.DECISION.MSC ROUTINE AT TIME.V=  241.706
     MSG.NO   23001   FROM   992456  TO   992200  TEXT  1091864  PHARCW  1  ORDER  0  T   992456  CRIT   0            RETURN  3
     ENTERED MESSAGE GENERATOR AT TIME.V =    243.904
     MSG.NO   20002  OF TYPE    0  ON NET   4  FROM TANK  20  TO TANK  19                  +++  GENERATED   +++

     END.XSMN EVENT ENTERED AT TIME.V=  271.706
     MSG.NO   20001  OF TYPE   20  ON NET   4  FROM TANK  20  TO TANK  24                  +++  END.XSMN COMPLETED   +++

     UNIT 41 IS RESTRICTED AND ORDERED THE REQUESTING CC NOT TO MOVE AT    271.706
******EXECUTING CCN.MOVE.DECISION.MSC ROUTINE AT TIME.V=  271.706
     MSG.NO   24002   FROM   992200  TO   992456  TEXT  1092096  PHARCW  1  ORDER  0  T   992456  CRIT   1            RETURN 14
     ENTERED MESSAGE GENERATOR AT TIME.V =    298.512
     MSG.NO    6001  OF TYPE    0  ON NET   2  FROM TANK   6  TO TANK   8                  +++  GENERATED   +++
     CALLING DECISION AT TIME.V=  322.349  FCN= 1  CCL= 1  J=  5  T=  993416
22222-----STATUS OF ACFR=  2  OF PFARCW=  1 CAUSED A RETURN

     END.XSMN EVENT ENTERED AT TIME.V=  326.512
     MSG.NO    6001  OF TYPE    0  ON NET   2  FROM TANK   6  TO TANK   8                  +++  END.XSMN COMPLETED   +++

     ENTERED MESSAGE GENERATOR AT TIME.V =    341.010
     MSG.NO   10001  OF TYPE    0  ON NET   2  FROM TANK  10  TO TANK   6                  +++  GENERATED   +++
     END.XSMN EVENT ENTERED AT TIME.V=  368.575
     MSG.NO   24002  OF TYPE   14  ON NET   4  FROM TANK  24  TO TANK  20                  +++  END.XSMN COMPLETED   +++

     COMPANY 20 HAS BEEN ORDERED NOT TO MOVE

     ENTERED MESSAGE GENERATOR AT TIME.V =    374.117
     MSG.NO   22001  OF TYPE    0  ON NET   2  FROM TANK  22  TO TANK   9                  +++  GENERATED   +++
```

```
END.XSMN EVENT ENTERED AT TIME.V= 1334.136
MSG.NO   3001  OF TYPE  26  ON NET   1  FROM TANK   3  TO TANK  21                +++   END.XSMN COMPLETED   +++

ONE IS NOT RESTRICTED AND ORDERED COMPANY NET TO MOVE AT  1334.136

*****EXECUTING GEN.MOVE.DECISION.MSC ROUTINE AT TIME.V= 1334.136
    MSG.NO  21002   FROM   992352  TO    993544  TEXT  1091224  PHAROW  1  ORDER  0  T    993544  CRIT   1              RETURN 12

    ENTERED MESSAGE GENERATOR AT TIME.V =  1347.135
    MSG.NO  15002  OF TYPE   C  ON NET   3  FROM TANK  15  TO TANK  13                +++   GENERATED   +++

    END.XSMN EVENT ENTERED AT TIME.V= 1350.044
    MSG.NO  11003  OF TYPE   C  ON NET   3  FROM TANK  11  TO TANK  13                +++   END.XSMN COMPLETED   +++

    CALLING DECISION AT TIME.V= 1376.457  RCW= 2  CCL= 2  J=  2  T=  993608
22222-----STATUS OF ACFR=  2  OF PHAROW=  1 CAUSED A RETURN
    END.XSMN EVENT ENTERED AT TIME.V= 1377.044
    MSG.NO  14003  OF TYPE   C  ON NET   3  FROM TANK  14  TO TANK  11                +++   END.XSMN COMPLETED   +++

    ENTERED MESSAGE GENERATOR AT TIME.V =  1389.045
    MSG.NO  21003  OF TYPE   G  ON NET   5  FROM TANK  21  TO TANK  24                +++   GENERATED   +++

    END.XSMN EVENT ENTERED AT TIME.V= 1400.278
    MSG.NO  21002  OF TYPE  14  ON NET   1  FROM TANK  21  TO TANK   3                +++   END.XSMN COMPLETED   +++

    COMPANY  3 HAS BEEN ORDERED NOT TO MOVE


    ENTERED MESSAGE GENERATOR AT TIME.V =  1421.235
    MSG.NO   9001  OF TYPE   C  ON NET   2  FROM TANK   9  TO TANK   6                +++   GENERATED   +++

    CALLING DECISION AT TIME.V= 1444.351  RCW= 5  CCL= 1  J=  3  T=  993544
11111-----CRESSTICU(I)) LE 1 CAUSED A RETURN
    END.XSMN EVENT ENTERED AT TIME.V= 1447.044
    MSG.NO  12004  OF TYPE   C  ON NET   3  FROM TANK  12  TO TANK  11                +++   END.XSMN COMPLETED   +++

    END.XSMN EVENT ENTERED AT TIME.V= 1455.045
    MSG.NO  21003  OF TYPE   G  ON NET   5  FROM TANK  21  TO TANK  24                +++   END.XSMN COMPLETED   +++

    ENTERED MESSAGE GENERATOR AT TIME.V =  1481.242
    MSG.NO  15003  OF TYPE   C  ON NET   3  FROM TANK  15  TO TANK  23                +++   GENERATED   +++

    END.XSMN EVENT ENTERED AT TIME.V= 1521.044
    MSG.NO  15002  OF TYPE   C  ON NET   3  FROM TANK  15  TO TANK  13                +++   END.XSMN COMPLETED   +++

    CALLING DECISION AT TIME.V= 1527.090  RCW= 2  CCL= 1  J=  6  T=  993352

*****EXECUTING GEN.MOVE.DECISION.MSC ROUTINE AT TIME.V= 1527.090
    MSG.NO   6004   FROM   993352  TO    992328  TEXT  1091224  PHAROW  1  ORDER  1  T    993352  CRIT   1              RETURN  1
    *** DECISION BASED ON RANGE TO ENEMY ******
    ENTERED MESSAGE GENERATOR AT TIME.V =  1536.509
    MSG.NO  22005  OF TYPE   C  ON NET   5  FROM TANK  22  TO TANK  25                +++   GENERATED   +++

    ENTERED MESSAGE GENERATOR AT TIME.V =  1564.393
    MSG.NO  13004  OF TYPE   C  ON NET   3  FROM TANK  13  TO TANK  11                +++   GENERATED   +++

    END.XSMN EVENT ENTERED AT TIME.V= 1596.509
    MSG.NO  22005  OF TYPE   C  ON NET   5  FROM TANK  22  TO TANK  25                +++   END.XSMN COMPLETED   +++

    CALLING DECISION AT TIME.V= 1604.784  RCW= 2  CCL= 2  J= 12  T=  992568

*****EXECUTING GEN.MOVE.DECISION.MSC ROUTINE AT TIME.V= 1604.784
    MSG.NO  12005   FROM   992568  TO    992264  TEXT  1091160  PHAROW  1  ORDER  0  T    992568  CRIT   1              RETURN  3

    ENTERED MESSAGE GENERATOR AT TIME.V =  1621.323
    MSG.NO  14004  OF TYPE   C  ON NET   3  FROM TANK  14  TO TANK  13                +++   GENERATED   +++
```

```
1  BDENT       5  BDECOM      0  BDELC       6  BLEGO       9  BRDE        1  BDERFCST    0.  NC.BDE.UNIT 25
1  N.BRIGADE   4  M.DIVISION  1

ATTRIBUTES OF EACH DIV.COMMANDER
1  DIV.T       5  DIVCOM      0  DIVLO       5  DIVGO       7  CDIV        1  DIVRFCST    0  NC.DIV.UNIT 26
1  N.DIVISION  1

    ENTERED MESSAGE GENERATOR AT TIME.V =  2029.182
    MSG.NC  21006  OF TYPE   C  ON NET   5  FROM TANK  21  TO TANK  26                +++  GENERATED   +++
    ENTERED MESSAGE GENERATOR AT TIME.V =  2087.123
    MSG.NC  22007  OF TYPE   C  ON NET   2  FROM TANK  22  TO TANK   6                +++  GENERATED   +++
    CALLING DECISION AT TIME.V= 2088.643  FCN= 4  CCL= 1  J=  3  T=  993544
*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 2088.643
    MSG.NC   3002   FROM   993544  TO   992392  TEXT  1090792  PHAROW  1  ORDER  1  T   993544  CRIT   1           RETURN  1
    +++ DECISION BASED ON RANGE TO ENEMY ++++++
    ENTERED MESSAGE GENERATOR AT TIME.V =  2109.291
    MSG.NC  24007  OF TYPE   C  ON NET   4  FROM TANK  24  TO TANK  18                +++  GENERATED   +++
    END.XSMN EVENT ENTERED AT TIME.V= 2112.244
    MSG.NC  13007  OF TYPE   C  ON NET   3  FROM TANK  13  TO TANK  12                +++  END.XSMN COMPLETED   +++

    END.XSMN EVENT ENTERED AT TIME.V= 2115.062
    MSG.NC  22006  OF TYPE  3C  ON NET   5  FROM TANK  22  TO TANK  25                +++  END.XSMN COMPLETED   +++
    BDET 1) IS RESTRICTED AND ORDERED THE REQUESTING BN NOT TO MOVE AT  2115.06
*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 2115.062
    MSG.NC  25003   FROM   992136  TO   992328  TEXT  1090840  PHAROW  1  ORDER  0  T   992328  CRIT   2           RETURN 20
    END.XSMN EVENT ENTERED AT TIME.V= 2143.024
    MSG.NC  24007  OF TYPE   C  ON NET   4  FROM TANK  24  TO TANK  18                +++  END.XSMN COMPLETED   +++

    END.XSMN EVENT ENTERED AT TIME.V= 2148.096
    MSG.NC  22007  OF TYPE   C  ON NET   2  FROM TANK  22  TO TANK   6                +++  END.XSMN COMPLETED   +++

    ENTERED MESSAGE GENERATOR AT TIME.V =  2162.390
    MSG.NC  22008  OF TYPE   C  ON NET   5  FROM TANK  22  TO TANK  21                +++  GENERATED   +++
    CALLING DECISION AT TIME.V= 2174.404  FCN= 7  CCL= 2  J= 16  T=  992712
33333-----STATUS OF MVANYTRCK=    7,CCL=    2) CAUSED EXIT WITH CRDEF=YES

    END.XSMN EVENT ENTERED AT TIME.V= 2174.665
    MSG.NC  24008  OF TYPE   C  ON NET   5  FROM TANK  24  TO TANK  22                +++  END.XSMN COMPLETED   +++

    ENTERED MESSAGE GENERATOR AT TIME.V =  2187.523
    MSG.NC   7002  OF TYPE   C  ON NET   2  FROM TANK   7  TO TANK  10                +++  GENERATED   +++
    ENTERED MESSAGE GENERATOR AT TIME.V =  2211.535
    MSG.NC  25004  OF TYPE   C  ON NET   5  FROM TANK  25  TO TANK  26                +++  GENERATED   +++
    END.XSMN EVENT ENTERED AT TIME.V= 2220.616
    MSG.NC  13005  OF TYPE   C  ON NET   3  FROM TANK  13  TO TANK  14                +++  END.XSMN COMPLETED   +++

    END.XSMN EVENT ENTERED AT TIME.V= 2236.019
    MSG.NC  22008  OF TYPE   C  ON NET   5  FROM TANK  22  TO TANK  21                +++  END.XSMN COMPLETED   +++
```

```
ENTERED MESSAGE GENERATOR AT TIME.V =  4018.380
MSG.NO  12008  OF TYPE   0  ON NET   3  FROM TANK  12  TO TANK  14          +++  GENERATED  +++

CALLING DECISION AT TIME.V= 4026.949  FC#= 2  CCL= 1  J= 15  T=  992776

######EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 4026.949
  MSG.NO  15013   FROM   992776  TO   992264  TEXT  1089928  PHASE  1  ORDER  1  T   952776  CRIT  1          RETURN  2
  +++ DECISION BASED ON RANGE TO ENEMY ++++++
```

ATTRIBUTES OF EACH COMPANY.COMMANDER

	CO.T	CRECST	COMPY	N.COMP.UNIT	M.BATTALION
1	2	0	1	1	1
2	2	0	2	1	1
3	3	1	3	1	1
4	3	0	4	1	1
5	3	0	5	1	1
6	3	0	6	1	1
7	6	0	7	1	1
8	6	1	8	1	1
9	2	0	9	1	1
10	2	1	10	1	1
11	1	0	11	1	1
12	1	1	12	1	1
13	1	0	13	1	1
14	1	0	14	1	1
15	2	0	15	1	1
16	5	0	16	1	1
17	5	0	17	1	1
18	6	0	18	1	1
19	7	0	19	1	1
20	2	1	20	1	1

ATTRIBUTES OF EACH BN.COMMANDER

	BN.T	BNCOR	BNLO	BNGO	BATT	BRECST	NO.BN.UNIT
1	3	3	4	6	1	0	21
2	4	8	2	3	2	1	22
3	3	1	6	8	3	0	23
4	3	2	3	6	4	0	24

	M.BRIGADE	N.BATTALION
1	1	5
2	1	5
3	1	5
4	1	5

ATTRIBUTES OF EACH BDE.COMMANDER

	BDE.T	BDECOR	BDELO	BDEGO	BRDE	BDERECST	NO.BDE.UNIT
1	5	4	6	9	1	0	25

	N.BRIGADE	M.DIVISION
1	4	1

ATTRIBUTES OF EACH DIV.COMMANDER

	DIV.T	DIVCOR	DIVLO	DIVGO	DDIV	DIVRECST	NO.DIV.UNIT
1	5	0	5	7	1	0	26

	N.DIVISION
1	1

```
END.XSMN EVENT ENTERED AT TIME.V= 4038.030
MSG.NO  15011  OF TYPE   0  ON NET   3  FROM TANK  15  TO TANK  11          +++  END.XSMN COMPLETED  +++
```

```
     MSG.NO  17005  OF TYPE   0  ON NET   4  FROM TANK  17  TO TANK  16                  +++   GENERATED   +++

     END.ASMN EVENT ENTERED AT TIME.V= 4331.532
     MSG.NO   6003  OF TYPE   0  ON NET   2  FROM TANK   6  TO TANK   8                  +++   END.XSMN COMPLETED   +++

     CALLING DECISION AT TIME.V= 4344.677  RCN= 7  CCL= 1  J= 16  T=  992712

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 4344.677
     MSG.NO  16004   FROM   992712  TO    992200  TEXT  1089816  PHARCH  1  ORDER  1  T   992712  CRIT   1             RETURN  1
     *** DECISION BASED ON RANGE TO ENEMY ******

     ENTERED MESSAGE GENERATOR AT TIME.V =  4357.191
     MSG.NO  12010  OF TYPE   0  ON NET   3  FROM TANK  12  TO TANK  15                  +++   GENERATED   +++

     END.ASMN EVENT ENTERED AT TIME.V= 4362.263
     MSG.NO  24012  OF TYPE  30  ON NET   5  FROM TANK  24  TO TANK  25                  +++   END.XSMN COMPLETED   +++

     BEEF II IS RESTRICTED AND IS REQUESTING PERMISSION TO MOVE AT  4362.28300

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 4362.283
     MSG.NO  25000   FROM   992136  TO    992072  TEXT  1089952  PHARCH  1  ORDER  0  T   992136  CRIT   2             RETURN 19

     ENTERED MESSAGE GENERATOR AT TIME.V =  4394.735
     MSG.NO  22013  OF TYPE   0  ON NET   5  FROM TANK  22  TO TANK  25                  +++   GENERATED   +++

     END.ASMN EVENT ENTERED AT TIME.V= 4421.555
     MSG.NO  16004  OF TYPE  21  ON NET   4  FROM TANK  16  TO TANK  24                  +++   END.XSMN COMPLETED   +++
44444*******OD    992712 NOTIFIED BY    992200 BY MESSAGE (TEXT= 1089816) THAT IT IS MOVING FROM ITS CURRENT POSITION.

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 4421.555
     MSG.NO  16005   FROM   992712  TO    992136  TEXT  1089904  PHARCH  1  ORDER  1  T   992200  CRIT   1             RETURN 16

     ENTERED MESSAGE GENERATOR AT TIME.V =  4427.765
     MSG.NO  25007  OF TYPE   0  ON NET   5  FROM TANK  25  TO TANK  22                  +++   GENERATED   +++

     CALLING DECISION AT TIME.V= 4428.565  RCN= 1  CCL= 1  J= 12  T=  992588
11111-----CROSSTICCLITT EQ 1 CAUSED A RETURN

     END.ASMN EVENT ENTERED AT TIME.V= 4435.735
     MSG.NO  22013  OF TYPE   0  ON NET   5  FROM TANK  22  TO TANK  25                  +++   END.XSMN COMPLETED   +++

     ENTERED MESSAGE GENERATOR AT TIME.V =  4473.557
     MSG.NO  19003  OF TYPE   0  ON NET   4  FROM TANK  19  TO TANK  17                  +++   GENERATED   +++

     END.ASMN EVENT ENTERED AT TIME.V= 4478.735
     MSG.NO  25007  OF TYPE   0  ON NET   5  FROM TANK  25  TO TANK  22                  +++   END.XSMN COMPLETED   +++

     CALLING DECISION AT TIME.V= 4493.251  RCN= 6  CCL= 1  J= 14  T=  992840

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 4493.251
     MSG.NO  14007   FROM   992840  TO    992264  TEXT  1089792  PHARCH  1  ORDER  1  T   992840  CRIT   1             RETURN  1
     *** DECISION BASED ON RANGE TO ENEMY ******

     ENTERED MESSAGE GENERATOR AT TIME.V =  4494.259
     MSG.NO  23012  OF TYPE   0  ON NET   5  FROM TANK  23  TO TANK  24                  +++   GENERATED   +++

     END.ASMN EVENT ENTERED AT TIME.V= 4456.557
     MSG.NO  19003  OF TYPE   0  ON NET   4  FROM TANK  19  TO TANK  17                  +++   END.XSMN COMPLETED   +++

     ENTERED MESSAGE GENERATOR AT TIME.V =  4515.465
     MSG.NO  24015  OF TYPE   0  ON NET   5  FROM TANK  24  TO TANK  21                  +++   GENERATED   +++

     END.ASMN EVENT ENTERED AT TIME.V= 4522.597
     MSG.NO  17005  OF TYPE   0  ON NET   4  FROM TANK  17  TO TANK  16                  +++   END.XSMN COMPLETED   +++

     END.ASMN EVENT ENTERED AT TIME.V= 4548.255
     MSG.NO  23012  OF TYPE   0  ON NET   5  FROM TANK  23  TO TANK  24                  +++   END.XSMN COMPLETED   +++

     CALLING DECISION AT TIME.V= 4556.535  RCN= 7  CCL= 1  J=   5  T=  993416
22222-----STATUS OF ACMR=  2  GT PHARCH=  1 CAUSED A RETURN
```

```
ENTERED MESSAGE GENERATOR AT TIME.V =  4576.043
MSG.NO  22014  OF TYPE   C  ON NET   5  FROM TANK  22  TO TANK  24                  +++  GENERATED  +++

END.XSMN EVENT ENTERED AT TIME.V= 4576.259
MSG.NO  16005  OF TYPE  21  ON NET   5  FROM TANK  24  TO TANK  25                  +++  END.XSMN COMPLETED  +++

AAAAA+++++ON   992200 NOTIFIED BEE   992136 BY MESSAGE (TEXT= 1089904) THAT ONE OF ITS COMPANIES IS MOVING FROM ITS POSITION.
     CALLING DECISION AT TIME.V= 4622.267  RCW= 2  CCL= 2  J= 11  I=  993032

AAAAAEXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 4622.267
     MSG.NO  11006   FROM   993032  TO    992264  TEXT  1089904  PHARCW  1  ORDER  1  T   993032  CRIT   1          RETURN  1
     +++ DECISION BASED ON RANGE TO ENEMY ++++++

ENTERED MESSAGE GENERATOR AT TIME.V =  4624.223
MSG.NO  13005  OF TYPE   C  ON NET   3  FROM TANK  13  TO TANK  11                  +++  GENERATED  +++

ENTERED MESSAGE GENERATOR AT TIME.V =  4641.277
MSG.NO  14006  OF TYPE   C  ON NET   3  FROM TANK  14  TO TANK  12                  +++  GENERATED  +++

END.XSMN EVENT ENTERED AT TIME.V= 4680.223
MSG.NO  13005  OF TYPE   C  ON NET   3  FROM TANK  13  TO TANK  11                  +++  END.XSMN COMPLETED  +++

     CALLING DECISION AT TIME.V= 4684.605  RCW= 4  CCL= 2  J=  9  I=  993160

AAAAAEXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 4684.605
     MSG.NO   9004   FROM   993160  TO    992328  TEXT  1089728  PHARCW  1  ORDER  1  T   993160  CRIT   1          RETURN  1
     +++ DECISION BASED ON RANGE TO ENEMY ++++++

ENTERED MESSAGE GENERATOR AT TIME.V =  4695.529
MSG.NO  22015  OF TYPE   C  ON NET   2  FROM TANK  22  TO TANK  10                  +++  GENERATED  +++

END.XSMN EVENT ENTERED AT TIME.V= 4700.334
MSG.NO  24013  OF TYPE   C  ON NET   5  FROM TANK  24  TO TANK  21                  +++  END.XSMN COMPLETED  +++

END.XSMN EVENT ENTERED AT TIME.V= 4734.101
MSG.NO  14006  OF TYPE   C  ON NET   3  FROM TANK  14  TO TANK  12                  +++  END.XSMN COMPLETED  +++

ENTERED MESSAGE GENERATOR AT TIME.V =  4744.546
MSG.NO  24014  OF TYPE   C  ON NET   5  FROM TANK  24  TO TANK  25                  +++  GENERATED  +++
     CALLING DECISION AT TIME.V= 4745.748  RCW= 2  CCL= 1  J= 18  I=  992584

AAAAAEXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 4745.748
     MSG.NO  15005   FROM   992584  TO    992200  TEXT  1089016  PHARCW  1  ORDER  1  T   992584  CRIT   1          RETURN  1
     +++ DECISION BASED ON RANGE TO ENEMY ++++++

END.XSMN EVENT ENTERED AT TIME.V= 4746.709
MSG.NO  22012  OF TYPE   C  ON NET   2  FROM TANK  22  TO TANK   7                  +++  END.XSMN COMPLETED  +++

ENTERED MESSAGE GENERATOR AT TIME.V =  4755.656
MSG.NO  24015  OF TYPE   C  ON NET   4  FROM TANK  24  TO TANK  17                  +++  GENERATED  +++

END.XSMN EVENT ENTERED AT TIME.V= 4766.177
MSG.NO  25006  OF TYPE  40  ON NET   5  FROM TANK  25  TO TANK  26                  +++  END.XSMN COMPLETED  +++
     MOVE IS NOT RESTRICTED AND CAVE DOES PERMISSION TO MOVE AT  4766.177

AAAAAEXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 4766.177
     MSG.NO  25006   FROM   992136  TO    992136  TEXT  1089544  PHARCW  1  ORDER  0  T   992136  CRIT   5          RETURN 22

     CALLING DECISION AT TIME.V= 4808.274  RCW= 1  CCL= 2  J=  4  I=  993480

AAAAAEXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 4808.274
     MSG.NO   4001   FROM   993480  TO    992352  TEXT  1089472  PHARCW  1  ORDER  1  T   993480  CRIT   5          RETURN  1
     +++ DECISION BASED ON RANGE TO ENEMY ++++++

END.XSMN EVENT ENTERED AT TIME.V= 4810.101
MSG.NO  12010  OF TYPE   C  ON NET   3  FROM TANK  12  TO TANK  15                  +++  END.XSMN COMPLETED  +++

ENTERED MESSAGE GENERATOR AT TIME.V =  4817.401
MSG.NO   9005  OF TYPE   C  ON NET   2  FROM TANK   9  TO TANK  22                  +++  GENERATED  +++
```

```
END.XSMA EVENT ENTERED AT TIME.V= 4819.035
MSG.NO  24014  OF TYPE   C  ON NET   5  FROM TANK  24  TO TANK  25          +++   END.XSMA COMPLETED   +++

ENTERED MESSAGE GENERATOR AT TIME.V =  4851.570
MSG.NO  15014  OF TYPE   C  ON NET   3  FROM TANK  15  TO TANK  13          +++   GENERATED   +++

END.XSMA EVENT ENTERED AT TIME.V= 4852.709
MSG.NO  22015  OF TYPE   C  ON NET   2  FROM TANK  22  TO TANK  10          +++   END.XSMA COMPLETED   +++

END.XSMA EVENT ENTERED AT TIME.V= 4875.056
MSG.NO  24015  OF TYPE   C  ON NET   4  FROM TANK  24  TO TANK  17          +++   END.XSMA COMPLETED   +++

CALLING DECISION AT TIME.V= 4876.816  ROW= 3  COL= 2  J= 15  I=  992776
*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 4876.816
MSG.NO  15015  FROM   992776  TO   992264  TEXT  1089952  PMARCH  1  ORDER  1  T   992776  CRIT   5        RETURN  1
*** DECISION BASED ON RANGE TO ENEMY ******

ENTERED MESSAGE GENERATOR AT TIME.V =  4903.107
MSG.NO   6016  OF TYPE   C  ON NET   2  FROM TANK   6  TO TANK   9          +++   GENERATED   +++

END.XSMA EVENT ENTERED AT TIME.V= 4921.039
MSG.NO  22014  OF TYPE   C  ON NET   5  FROM TANK  22  TO TANK  24          +++   END.XSMA COMPLETED   +++

ENTERED MESSAGE GENERATOR AT TIME.V =  4938.882
MSG.NO  18016  OF TYPE   C  ON NET   4  FROM TANK  18  TO TANK  20          +++   GENERATED   +++

END.XSMA EVENT ENTERED AT TIME.V= 4945.412
MSG.NO   9005  OF TYPE   C  ON NET   2  FROM TANK   9  TO TANK  22          +++   END.XSMA COMPLETED   +++

CALLING DECISION AT TIME.V= 4945.424  ROW= 3  COL= 1  J= 16  I=  992712
*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 4945.424
MSG.NO  16006  FROM   992712  TO   992200  TEXT  1089408  PMARCH  1  ORDER  1  T   992712  CRIT   5        RETURN  1
*** DECISION BASED ON RANGE TO ENEMY ******

ENTERED MESSAGE GENERATOR AT TIME.V =  4966.822
MSG.NO  26005  OF TYPE   C  ON NET   5  FROM TANK  26  TO TANK  25          +++   GENERATED   +++

END.XSMA EVENT ENTERED AT TIME.V= 4988.882
MSG.NO  18016  OF TYPE   C  ON NET   4  FROM TANK  18  TO TANK  20          +++   END.XSMA COMPLETED   +++

ENTERED MESSAGE GENERATOR AT TIME.V =  5006.088
MSG.NO  19004  OF TYPE   C  ON NET   4  FROM TANK  19  TO TANK  17          +++   GENERATED   +++

CALLING DECISION AT TIME.V= 5021.000  ROW= 6  COL= 2  J= 18  I=  992584
33333-----STATUS OF MVAPPROACH=    8,CCL=    2) CAUSED EXIT WITH ORDER=YES

ENTERED MESSAGE GENERATOR AT TIME.V =  5026.829
MSG.NO  21011  OF TYPE   C  ON NET   5  FROM TANK  21  TO TANK  26          +++   GENERATED   +++

END.XSMA EVENT ENTERED AT TIME.V= 5037.149
MSG.NO  26005  OF TYPE   C  ON NET   5  FROM TANK  26  TO TANK  25          +++   END.XSMA COMPLETED   +++

END.XSMA EVENT ENTERED AT TIME.V= 5062.019
MSG.NO   6016  OF TYPE   C  ON NET   2  FROM TANK   6  TO TANK   9          +++   END.XSMA COMPLETED   +++

ENTERED MESSAGE GENERATOR AT TIME.V =  5070.821
MSG.NO  18011  OF TYPE   C  ON NET   4  FROM TANK  18  TO TANK  16          +++   GENERATED   +++

ENTERED MESSAGE GENERATOR AT TIME.V =  5093.803
MSG.NO  16007  OF TYPE   C  ON NET   4  FROM TANK  16  TO TANK  17          +++   GENERATED   +++

END.XSMA EVENT ENTERED AT TIME.V= 5097.149
MSG.NO  25008  OF TYPE  33  ON NET   5  FROM TANK  25  TO TANK  25          +++   END.XSMA COMPLETED   +++

UNIT 13 NOT RESTRICTED AND CAVE BAS PERMISSION TO MOVE AT  5097.149
```

```
*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5097.149
     MSG.NO  25009   FROM   992136   TO   992392   TEXT  1089340  PHARCH  1  CRDER  0  T    992392  CRIT   3           RETURN  6

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5097.149
     MSG.NO  25010   FROM   992136   TO   992328   TEXT  1089216  PHARCH  1  CRDER  0  T    992328  CRIT   3           RETURN  6

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5097.149
     MSG.NO  25011   FROM   992136   TO   992264   TEXT  1089128  PHARCH  1  CRDER  0  T    992264  CRIT   3           RETURN  6

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5097.149
     MSG.NO  25012   FROM   992136   TO   992200   TEXT  1089040  PHARCH  1  CRDER  0  T    992200  CRIT   3           RETURN  6

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5097.149
     MSG.NO  25013   FROM   992136   TO   992072   TEXT  1088952  PHARCH  1  CRDER  0  T    992136  CRIT   3           RETURN 18
CCCCC*****BDE   992136 NOTIFIED DIV   992072 BY MESSAGE NO  25013 THAT IT HAS GIVEN ALL BATTALIONS PERMISSION TO MOVE AT WILL.

     CALLING DECISION AT TIME.V= 5110.440  RCW= 5  COL= 1  J= 15  T=  992776

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5110.440
     MSG.NO  25016   FROM   992776   TO   992264   TEXT  1089544  PHARCH  1  CRDER  1  T    992776  CRIT   3           RETURN  1
     *** DECISION BASED ON RANGE TO ENEMY ******

     END.XSMN EVENT ENTERED AT TIME.V= 5120.149
     MSG.NO  21011  OF TYPE    C  ON NET    5  FROM TANK  21  TO TANK  26                    ***   END.XSMN COMPLETED   ***

     ENTERED MESSAGE GENERATOR AT TIME.V =  5125.803
     MSG.NO  24016  OF TYPE    C  ON NET    4  FROM TANK  24  TO TANK  19                    ***   GENERATED   ***

     END.XSMN EVENT ENTERED AT TIME.V= 5139.992
     MSG.NO  18011  OF TYPE    C  ON NET    4  FROM TANK  18  TO TANK  16                    ***   END.XSMN COMPLETED   ***

     ENTERED MESSAGE GENERATOR AT TIME.V =  5142.582
     MSG.NO  10007  OF TYPE    C  ON NET    2  FROM TANK  10  TO TANK   8                    ***   GENERATED   ***

     ENTERED MESSAGE GENERATOR AT TIME.V =  5165.747
     MSG.NO  10008  OF TYPE    C  ON NET    2  FROM TANK  10  TO TANK   9                    ***   GENERATED   ***

     CALLING DECISION AT TIME.V= 5198.317  RCW= 4  COL= 2  J=  3  T=  993544

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5198.317
     MSG.NO   3003   FROM   993544   TO   992392   TEXT  1089104  PHARCH  1  CRDER  1  T    993544  CRIT   3           RETURN  1
     *** DECISION BASED ON RANGE TO ENEMY ******

     END.XSMN EVENT ENTERED AT TIME.V= 5217.992
     MSG.NO  24016  OF TYPE    C  ON NET    4  FROM TANK  24  TO TANK  19                    ***   END.XSMN COMPLETED   ***

     END.XSMN EVENT ENTERED AT TIME.V= 5222.459
     MSG.NO  25009  OF TYPE   23  ON NET    5  FROM TANK  25  TO TANK  21                    ***   END.XSMN COMPLETED   ***

     BN IS NOT RESTRICTED AND GAVE COS PERMISSION TO MOVE AT  5222.459

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5222.459
     MSG.NO  21012   FROM   992392   TO   993672   TEXT  1089016  PHARCH  1  CRDER  0  T    993672  CRIT   3           RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5222.459
     MSG.NO  21013   FROM   992392   TO   993608   TEXT  1088064  PHARCH  1  CRDER  0  T    993608  CRIT   3           RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5222.459
     MSG.NO  21014   FROM   992392   TO   993544   TEXT  1088720  PHARCH  1  CRDER  0  T    993544  CRIT   3           RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5222.459
     MSG.NO  21015   FROM   992392   TO   993480   TEXT  1088632  PHARCH  1  CRDER  0  T    993480  CRIT   3           RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5222.459
     MSG.NO  21016   FROM   992392   TO   993416   TEXT  1088544  PHARCH  1  CRDER  0  T    993416  CRIT   3           RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5222.459
     MSG.NO  21017   FROM   992392   TO   992136   TEXT  1088456  PHARCH  1  CRDER  0  T    992392  CRIT   3           RETURN 11
BBBBB*****BN    992392 NOTIFIED BDE   992136 BY MESSAGE NO  21017 THAT IT HAS GIVEN ALL COMPANIES PERMISSION TO MOVE AT WILL.

     END.XSMN EVENT ENTERED AT TIME.V= 5232.982
     MSG.NO  10007  OF TYPE    0  ON NET    2  FROM TANK  10  TO TANK   8                    ***   END.XSMN COMPLETED   ***

     ENTERED MESSAGE GENERATOR AT TIME.V =  5936.985
```

```
MSG.NO  15010  OF TYPE   0  ON NET   3  FROM TANK  13  TO TANK  23                 +++  GENERATED  +++

END.XSPN EVENT ENTERED AT TIME.V= 5259.626
MSG.NO  25011  OF TYPE  23  ON NET   5  FROM TANK  25  TO TANK  23                 +++  END.XSPN COMPLETED  +++

ONE 31 NOT RESTRICTED AND GAVE CCS PERMISSION TO MOVE AT  5259.626

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5259.626
  MSG.NO  23013   FROM   992264  TO   993032  TEXT  1089520  PHARCW  1  ORDER  0  T   993032  CRIT   3          RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5259.626
  MSG.NO  23014   FROM   992264  TO   992568  TEXT  1089192  PHARCW  1  ORDER  0  T   992568  CRIT   3          RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5259.626
  MSG.NO  23015   FROM   992264  TO   992904  TEXT  1088272  PHARCW  1  ORDER  0  T   992904  CRIT   3          RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5259.626
  MSG.NO  23016   FROM   992264  TO   992840  TEXT  1088184  PHARCW  1  ORDER  0  T   992840  CRIT   3          RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5259.626
  MSG.NO  23017   FROM   992264  TO   992776  TEXT  1088096  PHARCW  1  ORDER  0  T   992776  CRIT   3          RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5259.626
  MSG.NO  23018   FROM   992264  TO   992136  TEXT  1088008  PHARCW  1  ORDER  0  T   992264  CRIT   3          RETURN 11
*********ON   992264 NOTIFIED EEE   992136 BY MESSAGE NO  23018 THAT IT HAS GIVEN ALL COMPANIES PERMISSION TO MOVE AT WILL.

CALLING DECISION AT TIME.V= 5264.941  RCV= 4  CCL= 1  J= 20  T=  992456

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5264.941
  MSG.NO  23066   FROM   992456  TO   992200  TEXT  1088608  PHARCW  1  ORDER  1  T   992456  CRIT   3          RETURN  1
+++ DECISION BASED ON RANGE TO ENEMY ++++++

ENTERED MESSAGE GENERATOR AT TIME.V =  5270.555
MSG.NO  24017  OF TYPE   0  ON NET   4  FROM TANK  24  TO TANK  18                 +++  GENERATED  +++

END.XSPN EVENT ENTERED AT TIME.V= 5288.582
MSG.NO  10000  OF TYPE   0  ON NET   2  FROM TANK  10  TO TANK   9                 +++  END.XSPN COMPLETED  +++

END.XSPN EVENT ENTERED AT TIME.V= 5305.719
MSG.NO  16007  OF TYPE   0  ON NET   4  FROM TANK  16  TO TANK  17                 +++  END.XSPN COMPLETED  +++

ENTERED MESSAGE GENERATOR AT TIME.V =  5310.679
MSG.NO  22016  OF TYPE   0  ON NET   2  FROM TANK  22  TO TANK   9                 +++  GENERATED  +++

END.XSPN EVENT ENTERED AT TIME.V= 5329.886
MSG.NO  25012  OF TYPE  23  ON NET   5  FROM TANK  25  TO TANK  24                 +++  END.XSPN COMPLETED  +++

ONE 41 NOT RESTRICTED AND GAVE CCS PERMISSION TO MOVE AT  5329.886

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5329.886
  MSG.NO  24018   FROM   992200  TO   992712  TEXT  1089384  PHARCW  1  ORDER  0  T   992712  CRIT   3          RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5329.886
  MSG.NO  24019   FROM   992200  TO   992648  TEXT  1088432  PHARCW  1  ORDER  0  T   992648  CRIT   3          RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5329.886
  MSG.NO  24020   FROM   992200  TO   992584  TEXT  1088248  PHARCW  1  ORDER  0  T   992584  CRIT   3          RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5329.886
  MSG.NO  24021   FROM   992200  TO   992520  TEXT  1088336  PHARCW  1  ORDER  0  T   992520  CRIT   3          RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5329.886
  MSG.NO  24022   FROM   992200  TO   992456  TEXT  1089840  PHARCW  1  ORDER  0  T   992456  CRIT   3          RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5329.886
  MSG.NO  24023   FROM   992200  TO   992136  TEXT  1087720  PHARCW  1  ORDER  0  T   992200  CRIT   3          RETURN 11
*********ON   992200 NOTIFIED EEE   992136 BY MESSAGE NO  24023 THAT IT HAS GIVEN ALL COMPANIES PERMISSION TO MOVE AT WILL.

END.XSPN EVENT ENTERED AT TIME.V= 5334.550
MSG.NO  21013  OF TYPE  13  ON NET   1  FROM TANK  21  TO TANK   2                 +++  END.XSPN COMPLETED  +++

COMPANY  2 HAS BEEN GIVEN PERMISSION TO MOVE AT WILL

END.XSMN EVENT ENTERED AT TIME.V= 5346.585
```

```
END.XSMN EVENT ENTERED AT TIME.V= 5750.399
MSG.NO  25010  OF TYPE  23  ON NET   5  FROM TANK  25  TO TANK  22                   +++   END.XSMN COMPLETED   +++

DIVE 21 NOT RESTRICTED AND GAVE COS PERMISSION TO MOVE AT  5750.399

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5750.399
     MSG.NO  22017   FROM   992328  TO   993352  TEXT  1088184  PHAROW  1  ORDER  0  T   993352  CRIT   3          RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5750.399
     MSG.NO  22018   FROM   992328  TO   993288  TEXT  1088544  PHAROW  1  ORDER  0  T   993288  CRIT   3          RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5750.399
     MSG.NO  22019   FROM   992328  TO   993224  TEXT  1088632  PHAROW  1  ORDER  0  T   993224  CRIT   3          RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5750.399
     MSG.NO  22020   FROM   992328  TO   993160  TEXT  1089192  PHAROW  1  ORDER  0  T   993160  CRIT   3          RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5750.399
     MSG.NO  22021   FROM   992328  TO   993096  TEXT  1089016  PHAROW  1  ORDER  0  T   993096  CRIT   3          RETURN 10

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5750.399
     MSG.NO  22022   FROM   992328  TO   992136  TEXT  1088520  PHAROW  1  ORDER  0  T   992328  CRIT   3          RETURN 11
&&&&&&&&&&&&BN    992328 NOTIFIED DOE   992136 BY MESSAGE NO  22022 THAT IT HAS GIVEN ALL COMPANIES PERMISSION TO MOVE AT WILL.

     END.XSMN EVENT ENTERED AT TIME.V= 5755.253
     MSG.NO  24025  OF TYPE   0  ON NET   4  FROM TANK  24  TO TANK  20                  +++   END.XSMN COMPLETED   +++

     END.XSMN EVENT ENTERED AT TIME.V= 5762.074
     MSG.NO  12012  OF TYPE   0  ON NET   3  FROM TANK  12  TO TANK  11                  +++   END.XSMN COMPLETED   +++

     ENTERED MESSAGE GENERATOR AT TIME.V =  5764.557
     MSG.NO   6011  OF TYPE   C  ON NET   2  FROM TANK   6  TO TANK   8                  +++   GENERATED   +++

     END.XSMN EVENT ENTERED AT TIME.V= 5798.557
     MSG.NO   6011  OF TYPE   G  ON NET   2  FROM TANK   6  TO TANK   8                  +++   END.XSMN COMPLETED   +++

     ENTERED MESSAGE GENERATOR AT TIME.V =  5802.244
     MSG.NO  24027  OF TYPE   C  ON NET   5  FROM TANK  24  TO TANK  22                  +++   GENERATED   +++

     END.XSMN EVENT ENTERED AT TIME.V= 5815.253
     MSG.NO  24022  OF TYPE  13  ON NET   4  FROM TANK  24  TO TANK  20                  +++   END.XSMN COMPLETED   +++

     COMPANY 20 HAS BEEN GIVEN PERMISSION TO MOVE AT WILL


     END.XSMN EVENT ENTERED AT TIME.V= 5816.495
     MSG.NO  25013  OF TYPE  41  ON NET   5  FROM TANK  25  TO TANK  26                  +++   END.XSMN COMPLETED   +++

     DIVE 11 NOT RESTRICTED AND GAVE DOES PERMISSION TO MOVE AT  5816.495

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5816.495
     MSG.NO  25017   FROM   992136  TO   992136  TEXT  1089840  PHAROW  1  ORDER  0  T   992136  CRIT   3          RETURN 22

     CALLING DECISION AT TIME.V= 5824.366  RCW= 6  CCL= 1  J= 16  I=  992712
33333-----STATUS OF MVARY(RCW=    8,CCL=    1) CAUSED EXIT WITH ORDER=YES

     ENTERED MESSAGE GENERATOR AT TIME.V =  5835.767
     MSG.NO   9006  OF TYPE   C  ON NET   2  FROM TANK   9  TO TANK  22                  +++   GENERATED   +++

     END.XSMN EVENT ENTERED AT TIME.V= 5836.074
     MSG.NO  11010  OF TYPE   C  ON NET   3  FROM TANK  11  TO TANK  12                  +++   END.XSMN COMPLETED   +++

     END.XSMN EVENT ENTERED AT TIME.V= 5861.253
     MSG.NO  17007  OF TYPE   C  ON NET   4  FROM TANK  17  TO TANK  19                  +++   END.XSMN COMPLETED   +++

     END.XSMN EVENT ENTERED AT TIME.V= 5871.839
     MSG.NO  24024  OF TYPE  31  ON NET   5  FROM TANK  25  TO TANK  26                  +++   END.XSMN COMPLETED   +++

     DIVE 11 NOT RESTRICTED AND GAVE DOES PERMISSION TO MOVE AT  5871.839

*****EXECUTING GEN.MOVE.DECISION.MSG ROUTINE AT TIME.V= 5871.839
```

BIBLIOGRAPHY

Broussard, G.J., A Dynamic Study of Factors Impacting on the Tank Commander's Target Selection Process, M.S. Thesis, Naval Postgraduate School, Monterey, CA, March 1979.

Caldwell, W.J. and Meiers, W.D., An Air to Ground and Ground to Air Combined Arms Combat Simulation (STAR-AIR), M.S. Thesis, Naval Postgraduate School Monterey, CA, September 1979.

Consolidated Analysis Center, Inc., Simscript II.5 Reference Handbook, March 1976.

Department of the Army, Review of Selected Army Models, May 1971.

General Research Corporation, OAD-CR-73, CARMONETTE, Volumes I, II, and III, November 1974.

Haislip, W.A., Communications/Electronic Warfare Modules for Simulation of Tactical Alternative Responses (STAR) Combat Model, M.S. Thesis, Naval Postgraduate School, Monterey, CA, June 1980.

Hartman, J.K., Ground Movement Modelling in the STAR Combat Model, Naval Postgraduate School, Technical Report NPS55-80-021, Monterey, CA, May 1980.

Hartman, J.K., Parametric Terrain and Line of Sight Modelling in the STAR Combat Model, Naval Postgraduate School, Technical Report NPS55-79-018, August 1979.

Kiviat, P.J., Villanueva, R., and Markowitz, H.M., SIMSCRIPT II.5 Programming Language, 2d Edition, Consolidated Analysis Center, Inc., 1973.

Kramer, J.S., Simulation of Dynamic Tactical Route Selection with Application in the STAR Model, M.S. Thesis, Naval Postgraduate School, Monterey, CA, March 1979.

Needels, C.J., Parameterization of Terrain in Army Combat Analysis, M. S. Thesis, Naval Postgraduate School, Monterey, CA, March 1976.

Parry, S.H., and Kelleher, E.P., Tactical Parameters and Input Requirements for the Ground Component of the STAR Combat Model, Naval Postgraduate School Technical Report NPS55-79-023, Monterey, CA, October 1979.

Starner, S.G., A Two-sided Field Artillery Stochastic Simulation, M.S. Thesis, Naval Postgraduate School, Monterey, CA, March 1979.

Thurman, E., and Carpenter, H., Simulation of Dismounted Infantry in the Simulation of Tactical Alternative Responses (STAR) Combat Model, M.S. Thesis, Naval Postgraduate School, Monterey, CA, June 1980.

Wallace, W.S. and Hagewood, E.G., Simulation of Tactical Alternative Responses (STAR), M.S. Thesis, Naval Postgraduate School, Monterey, CA, December 1978.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 55 Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
4.	Chief Tradoc Research Element Monterey Naval Postgraduate School Monterey, California 93940	1
5.	Major Philip A. Olson, Jr. 3110 Lakeview Dr. Leavenworth, Kansas 66048	1
6.	Associate Professor James K. Hartman, Code 55Hh Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
7.	Associate Professor S. H. Parry, Code55Py Department of Operations Research Naval Postgraduate School Monterey, California 93940	5
8.	Headquarters U. S. Army TRADOC ATTN: Director, Analysis Directorate (Col. D. Maddox) Ft. Monroe, Virginia 23651	1
9.	Associate Professor A. L. Schoenstadt, Code 53Zh Department of Mathematics Naval Postgraduate School Monterey, California 93940	10
10.	Office of the Commanding General U. S. Army TRADOC ATTN: General Donn A. Starry Ft. Monroe, Virginia 23651	1
11.	Headquarters U. S. Army Training & Doctrine Command ATTN: ATCG-T (Col. Scribner) Ft. Monroe, Virginia 23651	1

12. Headquarters 1
U. S. Army Training & Doctrine Command
ATTN: Director, Analysis Directorate
Combat Developments (MAJ Chris Needels)
Ft. Monroe, Virginia 23651

13. Headquarters 1
U. S. Army Training & Doctrine Command
ATTN: Director, Maneuver Directorate
Combat Developments (Col. Fred Franks)
Ft. Monroe, Virginia 23651

14. Mr. David Hardison 1
Deputy Under Secretary of the Army
(Operations Research)
Department of the Army, The Pentagon
Washington, D. C. 20310

15. LTG William Richardson 1
Commanding General
U. S. Army Combined Arms Center
Ft. Leavenworth, Kansas 66027

16. Director 1
Combined Arms Combat Development Activity
ATTN: Col. Reed
Ft. Leavenworth, Kansas 66027

17. Director, BSSD 1
Combined Arms Training Development Actiivity
ATTN: ATZLCA-DS
Ft. Leavenworth, Kansas 66027

18. Director 1
Combat Analysis Office
ATTN: Mr. Kent Pickett
U. S. Army Combined Arms Center
Ft. Leavenworth, Kansas 66027

19. Command and General Staff College 1
ATTN: Education Advisor
Room 123, Bell Hall
Ft. Leavenworth, Kansas 66027

20. Dr. Wilbur Payne, Director 1
U. S. Army TRADOC Systems Analysis Activity
White Sands Missile Range, New Mexico 88002

21. Headquarters, Department of the Army 1
Office of the Deputy Chief of Staff
for Operations and Plans
ATTN: DAMO-2D
Washington, D. C. 20310

22. Commander 1
U. S. Army Concepts Analysis Agency
ATTN: MOCA-SMS (CPT Steve Shupack)
8120 Woodmont Avenue
Bethesda, Maryland 20014

23. Commander 1
U. S. Army Concepts Analysis Agency
ATTN: MOCA-WG (LTC Earl Darden)
8120 Woodmont Avenue
Bethesda, Maryland 20014

24. Director 1
U. S. Army Night Vision & Electro-optical Lab.
ATTN: DEL-NV-VI (Mr. Frank Shields)
Ft. Belvoir, Virginia 22060

25. Director 1
U. S. Army Material Sustems Analysis Activity
ATTN: Mr. Will Brooks
Aberdeen Proving Grounds, Maryland 21005

26. Director 1
USATRASANA
ATTN: Mr. Ray Heath
White Sands Missile Range, New Mexico 88002

27. Commander, MICOM 1
ATTN: DRSMI-YC (Maj. Hagewood)
Redstone Arsenal, Alabama 35809

28. Col. Frank Day 1
TRADOC Systems Manager - XM1
U. S. Army Armor Center
Ft. Knox, Kentucky 40121

29. Director 1
Combat Developments, Studies Division
ATTN: MAJ W. Scott Wallace
U. S. Army Armor Agency
Ft. Knox, Kentucky 40121

30. Commandant 1
U. S. Army Field Artillery School
ATTN: ATSF-MBT (CPT Steve Starner)
Ft. Sill, Oklahoma 73503

31. Director, Combat Developments 1
ATTN: Col. Clark Burnett
U. S. Army Aviation Agency
Ft. Rucker, Alabama 36362

32. Director 1
Combat Developments
U. S. Army Infantry School
Ft. Benning, Georgia 31905

33. Director 1
Armored Combat Vehicle Technology Program
ATTN: Col. Fitzmorris
U. S. Army Armor Center
Ft. Knox, Kentucky 40121

34. Director 1
Combat Developments
ATTN: Maj. William D. Meiers
U. S. Army Air Defense Agency
Ft. Bliss, Texas 79905

35. Commander 1
U. S. Army Logistics Center
ATTN: ATCL-OS (Mr. Cammeron/CPT Schuessler)
Ft. Lee, Virginia 23801

36. Commander, USAMMCS 1
ATTN: ATSK-CD-CS (Mr. Lee/Mr. Marmon)
Redstone Arsenal, Alabama 35809

37. Commander 1
U. S. Army Combined Arms Center
ATTN: ATZL-CA-CAT (R. E. DeKinder, Jr.)
Ft. Leavenworth, Kansas 66027

38. Director 1
U. S. Army AMSAA
ATTN: DRXSY-AA (Mr. Tom Coyle)
Aberdeen Proving Grounds, Maryland 21005

39. Office of the Deputy Chief of Staff 1
for Combat Developments
U. S. Army TRADOC
ATTN: Major General Carl Vuono
Ft. Monroe, Virginia 23651

40. Deputy Commanding General 1
Combined Arms Combat Development Activity
ATTN: ATZL-CA-DC (MG Jack Walker)
Ft. Leavenworth, Kansas 66027

41. Commander 1
U. S. Army Logistics Center
ATTN: ATCL-CTD (Ltc. Arnold)
Ft. Lee, Virginia 23801

42. Director 1
Combat Arms Training and Doctrine Activity
Battle Simulations Directorate
ATTN: Ltc R.B. Kupisyewski
Ft. Leavenworth, Kansas 66027

43. Commandant 1
US Army Signal School
Ft. Gordon, Georgia 30905

44. Director 1
USA TRASANA
ATTN: ATAA-TCC (Maj. McDonnell)
White Sands Missile Range, New Mexico 88002

45. HQ, TRADOC 1
ATTN: ATCD-AS (Cpt. Haislip)
Ft. Monroe, Virginia 23651

46. HQ, Electronic Systems Command 1
ATTN: EWC/SATB (Mr. J. Ridgeway)
Kelly Air Force Base
San Antonio, Texas 78243

LMED

8